Adaptive active vision

Adaptive Active Vision

Adaptive Active Vision

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit Maastricht,
op gezag van de Rector Magnificus,
Prof. mr. G.P.M.F. Mols,
volgens het besluit van het College van Decanen,
in het openbaar te verdedigen
op donderdag 26 juni 2008 om 12.00 uur

door

Guido Cornelis Henricus Eugène de Croon

Promotores: Prof. dr. E.O. Postma
Prof. dr. H.J. van den Herik

Leden van de beoordelingscommissie:
1 Prof. dr. A.J. van Zanten (voorzitter)
2 Prof. dr. D.H. Ballard (University of Texas at Austin)
3 Prof. dr. R. Goebel
4 Prof. dr. A. Nijholt (Universiteit Twente)
5 Prof. dr. G. van Oortmerssen

This research has been funded by the Netherlands Organisation for Scientific Research (NWO), in the framework of the ToKeN project VINDIT (grant number 634.000.018).

 Dissertation Series No. 2008-18

The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

ISBN 978-90-7138-235-2

Cover design by G.C.H.E. de Croon, photos by N. Mooren.

Printed by Gildeprint, The Netherlands.

Preface

The vital role of visual actions in human and animal vision suggests that we may improve computer vision models by endowing them with visual actions as well. The main difficulty in creating such *active vision models* is to provide them with a step-by-step procedure for determining their actions. The cause of this difficulty is that we have insufficient knowledge of how humans and animals determine their visual actions.

In this thesis I study an approach to active vision that does not require the designer to predetermine what the model should do. Instead, the active vision model is *adapted* to a visual task. To this end, we use an *evolutionary algorithm*, which is inspired by biological evolution; by means of a 'survival of the fittest' it generates active vision models that are increasingly better at their task. The experiments in this thesis show that artificial evolution can result in rather surprising action strategies. The active vision models use these strategies to perform challenging visual tasks in a computationally efficient manner.

This thesis would not have been possible without the help of a large group of people. I would first like to thank my supervisors, Eric Postma and Jaap van den Herik. I am grateful to Eric for bringing order in the chaos of my research ideas, and for motivating me by being interested in almost every experiment that I undertook. I thank Jaap for his guidance in the writing process of the thesis and many of the articles on which it is based. He always succeeded in making me laugh about my own writing, which was the best way to make me remember how to improve it.

Furthermore, I would like to thank all people with whom I collaborated over the years. I am grateful to Ida Sprinkhuizen-Kuyper, who helped me with the comparison of different active vision models. Moreover, I will never forget my roommates over the four years: Laurens van der Maaten, Igor Berezhnoy, Niek Bergboer, Stijn Vanderlooy, and Sander Bakkes. I thank them for the moments of distraction and for being sceptical about my research subject; it helped me to develop and explain my research ideas. In particular, I would like to thank Laurens, since he took the effort to proofread almost the entire thesis. On this account, I would also like to thank Vito Trianni and Ben Torben-Nielsen, who proofread and commented on chapters of the thesis as well. In addition, I enjoyed my work and discussions with Michel van Dartel, Joyca Lacroix, Sander Spek, Steven de Jong, Marc Ponsen, Jahn Saito, Guillaume Chaslot, Andra Waagmeester, and Maarten Schadd. In my work I have received valuable support from Peter Geurtz, our system administrator, and from the secretariat: Joke Hellemons, Tons van den Bosch, and Liesbeth Nederlands. I

am grateful as well to all people with whom I collaborated and who made me feel at home at the EPFL in Lausanne, in particular to Mototaka Suzuki and Dario Floreano.

Furthermore, I am very much indebted to my friends for the good times over the years, and to my parents and brother for always being there for me. Finally, I have to admit that I am too little of a poet to express in a suitable way my gratitude to Bénédicte, my wife. She always brightens up my day, and hopefully will do so for many years to come.

Guido de Croon, March 2008.

Acknowledgements

This research has been funded by the Netherlands Organisation for Scientific Research (NWO), in the framework of the TOKEN project VINDIT (grant number 634.000.018). It has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems. Finally, I gratefully acknowledge financial support over the years by the Universiteitsfonds Limburg / SWOL.

Table of Contents

Chapter

1

Active Vision

Actions are essential to visual perception. Humans and animals influence what they see by moving their body, head, or eyes.

There are three main types of visual action in natural organisms. All three serve a different goal (Land and Nilsson, 2002). The first type of visual action is *gaze stabilisation*. The goal of gaze stabilisation is to keep the gaze fixed relative to the environment. This mostly involves slow eye movements to compensate for movements of the head or body (Easter, Johns, and Heckenlively, 1974; Paul, Nalbach, and Varjú, 1990; Schilstra and van Hateren, 1998; Land and Nilsson, 2002). Since the eyes are restricted in their movements, the slow movements are complemented by quick eye movements that re-centre the eye. Gaze stabilisation is mainly necessary because it takes time for optic signals to be integrated by the photoreceptors in the eye. When the eye makes a quick and large movement, blur occurs. The second type of visual action is *smooth pursuit*; it is used only by humans and primates (Land and Nilsson, 2002; Krauzlis, 2005). The goal of smooth pursuit is the tracking of a moving object. It involves smooth eye movements that employ predictions of the object's future positions (Barnes and Asselman, 1991). The third type of visual action is *intentional saccade*, which is common in humans and animals. The goal of an intentional saccade is to direct the gaze towards relevant elements in the environment. Intentional saccades are crucial for humans, because they have foveal vision: they possess a high resolution in the central two degrees of their visual field and an increasingly lower resolution outwards to the periphery (Anstis, 1973). Humans make on average three eye movements per second, in order to perceive different elements of interest at a high resolution (cf. Rao *et al.*, 1995).

So far, we neither understand how intentional saccades are determined, nor what information is extracted from the resulting visual inputs over time. We illustrate these two open questions by taking human eye movements as an example.

It is difficult to find out how humans determine their eye movements, since these movements are made on the basis of *implicit knowledge*. Humans have difficulty to explain how they determine their eye movements, and are often not even aware of them (Land and Hayhoe, 2001). One may wonder whether brain measurements could help us out. Current measurement techniques give information on the

brain areas that are involved in eye movements and on their respective functions (cf. Kranczioch *et al.*, 2005). However, they do not provide us with details on the underlying mechanisms that are in place during natural behaviour.

Most of the knowledge we have on the manner in which humans determine their eye movements has been obtained by means of behavioural studies. It is possible to gain insight into *what* eye-movements a human subject makes, by tracking his[1] eyes. Studies with eye-trackers have provided considerable insight into human gaze behaviour. Yarbus (1967) showed that eye movements are task dependent. The way humans look at a photo of a family when they are asked what the family is about to do, is different from when they are asked to search for a specific object. During the last decade it became possible to track eye movements of subjects while they perform everyday activities, such as making tea or a peanut butter sandwich. These studies have shown that eye movements greatly depend on the current demands of the task (Land and Hayhoe, 2001; Triesch *et al.*, 2003; Hayhoe and Ballard, 2005). In an overview of eye-movement studies, Henderson (2003) explained that eye movements also depend on episodic scene knowledge (our knowledge of the current situation) and on scene-schema knowledge (our assumptions of the world).

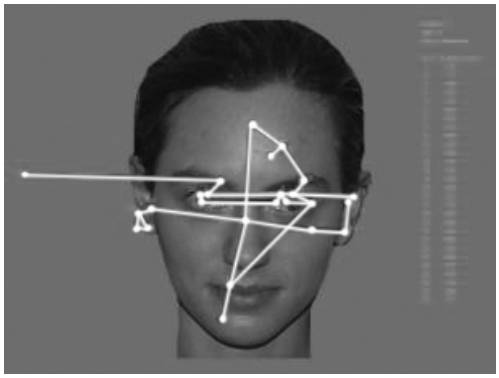


Figure 1.1: Example gaze trajectory of a subject that had to learn and recognise human faces. Dots represent fixations, and lines represent saccades. Reproduced with permission from: J.M. Henderson, R. Falk, S. Minut, F.C. Dyer, and S. Mahadevan (2001). Gaze control for face learning and recognition by humans and machines. *From Fragments to Objects: Segmentation Processes in Vision* (eds. T. Shipley and P. Kellman), pp. 463 - 481, Elsevier, New York, NY.

However, these eye-tracking studies do not tell us *how* humans determine what eye movements to make. The problem is that the studies only provide circumstantial evidence, such as evidence that humans tend to fixate conspicuous image locations (Rajashekar, Cormack, and Bovik, 2003; Rajashekar *et al.*, 2007). This prob-

[1] For brevity, we use 'he' and 'his' wherever 'he or she' and 'his or her' are meant.

lem is illustrated by Figure 1.1. It shows the eye-tracking data of a subject that had first to learn and subsequently to recognise human faces (Henderson *et al.*, 2001). The dots represent the subject's fixations, while the lines represent the saccades. The fixations of the subject seem to be on areas of the face that are useful for the recognition task, but they leave us with a number of questions. Here, we mention five such questions. Does each of the fixations contribute to the task? What information is extracted by the subject when he focuses on the forehead of the person to be recognised? Why does the subject make small gaze shifts when looking at the ears? And, importantly, how does the sequence of fixations arise? How relevant is the order of the fixations? At this moment, neither brain measurements nor behavioural studies can provide answers to these questions.

In this thesis we employ computational vision models in order to gain a better understanding of how visual actions can be determined, and what roles they can play in the vision process. Our research is part of the field of *Embodied Cognitive Science* (Varela, Maturana, and Uribe, 1974; Brooks, 1990; Clark, 1998; Pfeifer and Scheier, 1999), in which the use of computational active vision models is a common approach. Such models have the advantage that we have complete access to all internal processes that determine the visual actions (cf. Kortmann, 2001). By building computational active vision models, we simultaneously aim at two goals.

Our first goal is to obtain a better understanding of the general properties of the active vision process. Our models do not resemble any natural vision system in particular, but share two key properties with such systems. The first property is that our models are *situated* in an environment, which means that they can take actions that influence their future visual inputs by means of the environment. The second property is that our models are applied to visual tasks that natural organisms also have to cope with (classification, control, object detection). As a consequence, the models can improve our understanding of the use of actions for visual tasks that are relevant to natural organisms.

Our second goal is the improvement of computer vision models. Despite major advances in the last few decades, human vision still outperforms computer vision in many domains. The high performance of human vision is partly due to the use of visual actions. However, such actions play a negligible role in current state-of-the-art computer vision models. We believe that endowing them with visual actions may improve their performance as well.

This chapter is organised as follows. In Section 1.1, we discuss the field of active computer vision and a promising approach to constructing active vision models, named *adaptive active vision*. This leads us to our problem statement and four research questions, given in Section 1.2. We explain our research methodology in Section 1.3. Finally, we give an outline of the thesis in Section 1.4.

1.1 Active Computer Vision

The field of computer vision mainly focuses on *passive vision systems*. Such systems process incoming images to estimate the state of the world (Marr, 1982). However, based on the work by Gibson (1979), some later studies allow computer vi-

sion systems to take actions in their environment. These studies have shown that the use of actions can simplify visual tasks that are difficult in a passive setting. In an early paper on *active computer vision*[2] Aloimonos, Weiss, and Bandyopadhyay (1988) showed that actions can simplify the determination of shape and structure from visual cues, such as texture, shading, or motion. More recent studies have focused on how actions can facilitate classification and control tasks (Scheier, Pfeifer, and Kuniyoshi, 1998; Denzler and Brown, 2002; Nolfi and Marocco, 2002; Floreano *et al.*, 2004). Ballard (1991) argued that a foveal active vision system could be particularly advantageous for computer vision. A foveal active vision system processes only a part of the visual scene at a high resolution. Two main advantages of such a system would be that it can (1) simplify visual tasks by means of its actions, and (2) reduce the computational load, since only a small part of the visual scene has to be processed in detail.

The challenge in constructing an active vision model is to find an intelligent action strategy. This is difficult, because active vision is a process over time, in which incomplete observations and actions are tightly interrelated. One approach to the challenge is to make assumptions on the active vision process. For instance, many studies make the assumption that vision is an iterative process of state estimation and action selection. These studies remain close to the work by Marr (1982), since the resulting active vision models select actions that are useful to estimate the state of the world (i.e., state estimation). Subsequently, the estimated state can be used to take other, also non-visual, actions.

In our research we emphasise on creating active vision models, while making no assumptions on what action strategy the models should follow. The reason for this is that such assumptions may prevent the model from using strategies that may be vital for the success of humans and animals, but that are unknown to us. For this reason, we adopt the approach taken in the field of *Evolutionary Robotics* (Husbands *et al.*, 1997; Nolfi and Floreano, 2000). Studies in this field do not decompose robotic tasks into state estimation and acting, but they attempt to achieve the goal by an artificial evolution of the robot controller as a whole. The choice of the action strategy is left to the adaptation process of the evolutionary algorithm.

Although there are other suitable ways to adapt a controller, an evolutionary algorithm conveniently combines three desirable properties. First, it is a semi-supervised optimisation method: we can apply it to problems of which we know the goal but are unaware of what actions the model should take. Second, as a result of the semi-supervised nature, an evolutionary algorithm can find non-greedy action strategies. Since it optimises the performance of the model on the final goal, the adapted model does not necessarily have to select greedy actions at each time step. Third, it can optimise multiple parts of the model at the same time. In many experiments we optimise the model's visual features and controller together, which may improve the performance on the task (see, e.g., Macinnes and Di Paolo, 2006).

[2]Note that active (computer) vision is sometimes also referred to as animate vision or purposive vision. 'Animate vision' was introduced by Ballard (1991) to avoid confusion between active vision as explained in the text and active sensors such as laser rangefinders. 'Purposive vision' was introduced by Aloimonos (1990), Rivlin and Aloimonos (1992). The term mainly emphasises that vision has a purpose and that an active vision model should only use the representations that are necessary for its tasks.

In spite of the fact that we exclusively use evolutionary algorithms to optimise controllers, we do not consider it necessary to use a specific optimisation algorithm. Instead, we consider it most important that the active vision models do not incorporate assumptions on what action strategy to follow. As a consequence, the *adaptation* to the task becomes essential. Therefore, we prefer to use the term *Adaptive Active Vision*. This term groups studies that are different in optimisation algorithm but similar in spirit (e.g., Schmidhuber, 1990; Floreano *et al.*, 2004).

Early studies on adaptive active vision focused on proofs of principle. In Schmidhuber (1990), an active vision model had to learn to find a black triangle in white images corrupted by various amounts of noise. Harvey, Husbands, and Cliff (1994) applied an evolutionary algorithm to a robot in the real world that had to approach a triangle and avoid a rectangle, both drawn on a wall of the arena in which it moved around. Kato and Floreano (2001) investigated a rather similar task, but in static images: an active vision model had to differentiate black squares and triangles in images corrupted by various amounts of noise. In later work, adaptive active vision models were applied to more complex tasks such as wall avoidance (Marocco and Floreano, 2002), simulated driving (Floreano *et al.*, 2004), and landmark navigation (Suzuki and Floreano, 2006a; Suzuki and Floreano, 2006b).

Currently, there are three main obstacles on the way towards a successful application of adaptive active vision models in computer vision. First, it is uncertain whether such models can cope with visually challenging tasks. Second, the manner in which the models handle visual tasks is poorly understood. Third, active vision is always associated with servo-motor systems. It is not clear whether adaptive active vision models could also improve computer vision involving sets of static images. We remark that such sets are important for the success of a computer vision model, because they often occur in real-world applications (e.g., images on the internet).

1.2 Problem Statement and Research Questions

In this thesis, we investigate whether we can remove the three obstacles and arrive at a successful application of adaptive active vision models. For this purpose, we formulate the following problem statement.

> **Problem statement**: *How do adaptive active vision models handle challenging visual tasks?*

A visual task is challenging if humans are currently still better at it than state-of-the-art computer vision models. In most of the challenging tasks studied, we use a foveal active vision model, which processes only local image samples. The reason for this is that we believe such a model to be most promising for the field of computer vision; it is computationally efficient and can be applied to static images. A model with foveal vision has to direct its *gaze* to elements of interest in the visual scene. Therefore, we refer to such a model as a *gaze control model*. The gaze shifts of a gaze control model applied to static images may be interpreted as internal attention shifts (*covert attention*), or as visual actions (*overt attention*), cf. Findlay and

Gilchrist (2003). We do not consider this distinction important to our objectives, and will interpret the gaze shifts as visual actions.

To gain a good insight into the problem statement we focus on four main research questions (RQs).

RQ 1: How does the adaptive active vision approach relate to other approaches of active vision?

RQ 2: How does a memoryless adaptive gaze control model handle an image classification task?

RQ 3: How does an adaptive gaze control model use its gaze shifts in a control task?

RQ 4: Can an adaptive gaze control model perform on a par with state-of-the-art computer vision models on the task of object detection?

1.3 Research Methodology

We address the problem statement and the four research questions by employing the following research methodology.

We first perform a literature research to identify existing active vision models. We compare these models to each other in order to gain insight into their differences, both empirically and on the basis of their theoretical properties. This comparison helps us to answer the first research question (RQ 1).

Subsequently, we develop a framework for gaze control models. We adapt gaze control models that instantiate the framework to different challenging visual tasks. After adaptation, we analyse how they handle their task. The analysis contributes to answering the problem statement, and RQ 2 and 3 in particular. To answer RQ 4, we compare the performance of the model with that of state-of-the-art computer vision models. The research methodology is summarised in Figure 1.2.

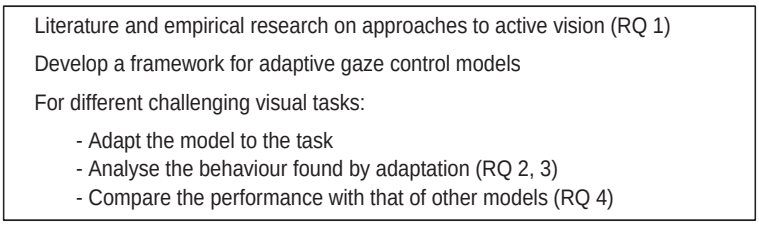


Figure 1.2: Overview of the research methodology used.

Four Tasks

For the execution of our research according to the methodology given above, we selected four tasks. Below, we list the tasks and the corresponding instantiations studied in the thesis.

1. **3-D object classification:** turn-table task (RQ 1)
2. **Image classification:** gender recognition (RQ 2)
3. **Control:** simulated driving task (RQ 3)
4. **Object detection:** face detection and car detection (RQ 4)

We have two main reasons for selecting the tasks listed above. First, the tasks are suitable for answering the corresponding research questions. For example, we selected the task of classifying 3-D objects to answer RQ 1, since all models can be applied to it without model changes.

Second, the tasks all impose different requirements on the gaze control model. The different requirements lead to different gaze strategies. If we understand these different strategies, we gain a broader comprehension of the problem statement. The tasks are mainly different in the following way. The classification and object detection tasks (RQ 1, 2, and 4) are *state-estimation tasks*, while the simulated driving task (RQ 3) is a *control task*. The difference between classification (RQ 1 and 2) and object detection (RQ 4) lies in how the active vision model employs *epistemic* and *pragmatic* actions (Kirsh and Maglio, 1994). Epistemic actions solely serve to gather information from the environment. Pragmatic actions aim to achieve a behavioural goal. In the classification tasks, the active vision model can use its visual actions entirely to gather information. In other words, the model employs solely epistemic actions. In the object-detection task, the active vision model has to direct its gaze towards objects in the image. As a consequence, its visual actions do not only have as goal to gather information but also to go to certain positions in the image. Therefore, the gaze control model has to balance epistemic and pragmatic actions.

1.4 Thesis Outline

In the thesis we study the problem statement and research questions introduced above. The remainder of the thesis is outlined as follows. In Chapter 2 we give an overview of existing active vision models and compare them with each other empirically on a task of 3-D object classification (RQ 1). Subsequently, in Chapter 3, we introduce a framework for our adaptive gaze control models. In Chapter 4 we analyse how an adaptive gaze control model handles a task of gender recognition in static natural images (RQ 2). Then, in Chapter 5, we analyse how an adaptive active vision model uses its gaze shifts to control a simulated car for a driving task (RQ 3). In Chapter 6, we apply an adaptive gaze control model to two object-detection tasks (face detection and car detection) and compare it with state-of-the-art object-detection methods (RQ 4). Thereafter, in Chapter 7, we discuss the findings of

the thesis on a general level. In Chapter 8 we provide our conclusion and give recommendations for further research.

Table 1.1 summarises which tasks and research questions are studied in the experimental chapters. Even though the chapters primarily focus on the research questions indicated in Table 1.1, they usually also give insight into some of the other questions. For example, in Chapter 4, we study how an active gaze control model handles a gender-classification task. However, our explanation of the model's action strategy is given in such a way that we may also compare it to other active vision models.

Table 1.1: Research questions addressed by the different experimental chapters.

	Task	RQ 1	RQ 2	RQ 3	RQ 4
Chapter 2	1	×			
Chapter 4	2		×		
Chapter 5	3			×	
Chapter 6	4				×

Chapter

2

Comparison of Active Vision Models

This chapter is based on the following articles[1].

1. G.C.H.E. de Croon, I.G. Sprinkhuizen-Kuyper, and E.O. Postma (in submission). Comparison of Active Vision Models. *Submitted to Image and Vision Computing*.
2. G.C.H.E. de Croon, I.G. Sprinkhuizen-Kuyper, and E.O. Postma (2006c). Comparing active vision models. *MICC-IKAT Technical Report 06-02*, Universiteit Maastricht, Maastricht, the Netherlands.

In this chapter, we focus on RQ 1: *How does the adaptive active vision approach relate to other approaches of active vision?* To answer this question, we compare existing active vision models with each other.

In the literature, a myriad of active vision models has been proposed that select their actions in different ways. Many of these active vision models are designed for specific tasks. For example, there are active vision models specifically constructed for detecting edges (Kass, Witkin, and Terzopoulos, 1988), for controlling the gaze of a simulated fish (Terzopoulos and Rabie, 1997), and for detecting a particular object in a visual scene (Minut and Mahadevan, 2001).

There are also models that seem to be instances of a more general approach to active vision. In the literature, we discern two such approaches: the *probabilistic approach* (see, e.g., Denzler and Brown, 2002) and the *adaptive approach* (see, e.g., Nolfi and Marocco, 2002).

The probabilistic approach regards active vision as an iterative process of state estimation and action selection. We employ the term 'probabilistic', since the approach uses a probabilistic framework for action selection. The central goal of probabilistic models is to reduce uncertainty on a predetermined part of the world state (cf. Schiele and Crowley, 1998; Paletta, Prantl, and Pinz, 1998; Kröse and Bunschoten, 1999; Borotschnig *et al.*, 2000; Arbel and Ferrie, 2001; Denzler and Brown, 2002; Deinzer, Denzler, and Niemann, 2003; Deutsch *et al.*, 2004). Typical tasks studied in this field include the classification of 3-D objects, depth estimation, and object tracking.

[1]The author would like to thank his co-authors for their permission to use parts of the articles in this chapter.

The adaptive approach does not employ an explicit two-tier process of state estimation and acting. We employ the term 'adaptive', since the approach adapts a predetermined structure (such as a neural network) to optimise performance on a specific task (cf. Harvey *et al.*, 1994; Hornby *et al.*, 2000; Nolfi and Marocco, 2002; Beer, 2003; Floreano *et al.*, 2004; van Dartel *et al.*, 2005; de Croon, Postma, and van den Herik, 2006b; Suzuki, van der Blij, and Floreano, 2006). Although the choice of the structure and the optimisation method impose implicit restrictions on the possible action strategies, the goal is not to predetermine what the active vision model should do internally. Typical tasks studied in this field include behavioural classification tasks and control tasks. An example of a behavioural classification task is the task in van Dartel *et al.* (2005), in which an agent has to catch big objects and avoid small ones.

It is hard to compare different models from the two approaches on the basis of the literature alone. The comparison is hampered for mainly two reasons. The first reason is that the active vision models of the two approaches are devised and studied in two isolated research fields. We regard the fields as isolated, since the studies of the different fields do not refer to each other. The second reason is that the models have not been compared empirically, even not within a single approach. Most of the different existing models have only been compared with either a random action model or with a passive vision model.

For the above two reasons, it is unclear how different types of active vision models relate to each other. In this chapter we aim at shedding more light on this matter by comparing six different vision models: three probabilistic active vision models, one adaptive active vision model, and two benchmark models (a random action model and a passive vision model). In our comparison we focus on a classification task of 3-D objects. We empirically determine how the six models compare to each other, both with respect to the classification performance they achieve and to the number of actions they need for classification.

In Section 2.1 we introduce the notation used to describe the vision models. Then, in Section 2.2, we explain the belief state update that is used by all models. In Section 2.3 the six vision models are described. We explain the experimental setup in Section 2.4. The experimental results are shown in Section 2.5, and are analysed in Section 2.6. Subsequently, we discuss the results of our comparison in Section 2.7. Finally, we provide the chapter conclusion in Section 2.8.

2.1 Notation

Throughout the chapter we will use the following notation. By $p(X)$ we indicate a probability distribution over all elements of the set X. Where capital letters represent sets, small letters represent elements. Therefore $p(x)$ is the probability of a specific element $x \in X$. Since we discuss the active vision models in the context of a classification task, we use the following variables: O is the set of all possible observations, C the set of all possible classes, and A the set of all possible actions. A specific observation, class, and action are denoted by o, c, and a, respectively, where $o \in O$, $c \in C$, and $a \in A$. In our discussion of the models, we will assume O, C, and

A to be discrete, finite sets, but all models can be applied to continuous variables as well (see, e.g., Denzler and Brown, 2002). The active vision process extends itself over multiple discrete time steps. The time step before the model takes an action, or performs an observation, is indicated by 0. A possible maximum number of time steps is indicated by t. The current time step is represented by i. A bold letter with subscript i indicates a sequence until time step i. For example, $\mathbf{o}_i = \langle o_1, o_2, \ldots, o_i \rangle$ is the sequence of observations of the first i time steps. Similarly, the sequence of corresponding actions is $\mathbf{a}_i = \langle a_1, a_2, \ldots, a_i \rangle$.

The belief state at a time step i represents the probabilities of the classes, given the past observations and actions: $p(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})$. The probability of a particular class c at time step i is indicated by $p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})$. On the basis of the current belief state $p(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})$, the probabilistic active vision models determine an action[2] for the current time step, a_i, which results in an observation o_i. The action and observation allow the models to calculate the belief state for the next time step, $p(C \mid \mathbf{o}_i, \mathbf{a}_i)$. At time step $i = 1$, the belief state is equal to the prior distribution, i.e., $p(C \mid \mathbf{o}_0, \mathbf{a}_0) = p(C)$, since $\mathbf{o}_0 = \mathbf{a}_0 = \langle\rangle$. The belief state at time step $i = 2$ is determined by the belief state update rule, explained in the next section.

2.2 Belief State Update

Since we want to focus on the differences in action selection strategies between the different types of active vision models, we provide all models with the same belief state update rule. We select the recursive belief state update introduced by Denzler and Brown (2000), since it can be employed by all vision models (see Section 2.3).

The goal of the belief state update is to calculate the posterior probability distribution $p(C \mid \mathbf{o}_i, \mathbf{a}_i)$. Denzler and Brown (2000) rewrite this posterior distribution with the help of the assumption that an observation only depends on the class and the current action. This assumption is referred to as the *Markov assumption*, and it can be expressed as follows.

$$p(o_i \mid C, \mathbf{o}_{i-1}, \mathbf{a}_i) = p(o_i \mid C, a_i) \tag{2.1}$$

The Markov assumption is reasonable in the light of the experiments by Denzler and Brown (2000). They study an active vision model that has to classify different 3-D objects. Each action of the active vision model corresponds to an angle from which an object can be viewed. Since in their experimental setup any angle can be reached at any time step, the observation does indeed not depend on past observations and actions (we note that our task and experimental setup are similar). The Markov assumption in Equation 2.1 can be used to obtain the following equation[3].

[2]The expression 'select an action' may imply that there is a finite set of discrete actions between which the model can choose. The expression 'determine an action' does not specifically imply either a discrete or a continuous action domain. In the thesis, we will not make this difference between the two expressions, and will use them interchangeably.

[3]We show how to obtain the equation in Appendix A.1. For a further discussion of the Markov assumption in a different context, see Subsection 4.6.2.

$$p(C \mid \mathbf{o}_i, \mathbf{a}_i) = \frac{p(o_i \mid C, a_i)p(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})}{\sum_{c \in C} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})p(o_i \mid c, a_i)} \tag{2.2}$$

This formula can be used as a recursive belief state update. Namely, to calculate the new belief state via Equation 2.2, we need (1) the old belief state and (2) the observation probability distributions $p(O \mid c, a)$ for all classes and actions. These distributions have to be determined before application of the active vision model. The subscript i is dropped here, since it is assumed that the distributions are static. In this chapter, all vision models employ the belief state update of Equation 2.2.

2.3 Six Vision Models

Below, we briefly introduce the six vision models that are compared in this chapter. The bold letters in the text clarify the abbreviations of the models. Subsection 2.3.1 to 2.3.6 contain a more detailed description of all the models.

The first three vision models belong to the probabilistic approach. They all take actions with the goal of reducing the uncertainty in the belief state, but employ different action selection strategies. The first model (MI) calculates the expected usefulness of all actions with the help of **M**utual **I**nformation. Then, it selects the best one. The second model (L-ER) **L**earns how to take actions to achieve **E**ntropy **R**eduction in the belief state. It learns an action policy that maps the current belief state to an action. The third model (RK) makes a **R**an**K**ing of all actions for each class in advance. During execution, it selects the action that has the highest rank for the most probable class.

There is no existing model from the adaptive approach that can be directly compared to the probabilistic models above. As a consequence, we created an active vision model that adheres to a main characteristic of adaptive active vision models: that they are directly adapted on the basis of the performance. For this reason, we introduced a fourth active vision model (L-CP) that does have an explicit belief state, but **L**earns to take actions to achieve a high **C**lassification **P**erformance. We remark that L-CP is different from most adaptive active vision models, since such models either have an implicit belief state or no belief state at all.

Models five and six are benchmark models. The fifth active vision model (RA) is an active vision model that takes **R**andom **A**ctions. The sixth vision model (PA) is a **PA**ssive vision model, that cannot perform actions and has to base its state estimation on only one observation. Below, we describe in more detail the above-described vision models.

In the thesis, we distinguish three types of parameters: *model parameters*, *task parameters*, and *training parameters*, to denote parameters that apply to models, tasks, and training, respectively. During our discussion of the models, we will only mention the model parameters that are trained for a specific task.

2.3.1 Mutual Information (MI)

The first model, mutual information (MI), belongs to a group of active vision models that perform tentative belief state updates for all possible actions, before selecting one action for actual execution (cf. Borotschnig *et al.*, 1999; Denzler and Brown, 2000; Borotschnig *et al.*, 2000; Denzler and Brown, 2002; Deutsch *et al.*, 2004). We implemented the model introduced in Denzler and Brown (2002). It selects an action a_i that results in the maximal mutual information between the observations and classes, given the past observations and actions:

$$a_i = \text{argmax}_a I(C; O \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a), \tag{2.3}$$

where the mutual information $I(C; O \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)$ is given by the following formula.

$$I(C; O \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) = \tag{2.4}$$

$$\sum_{c \in C} \sum_{o \in O} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}) p(o \mid c, a) \log \left(\frac{p(o \mid c, a)}{\sum_{c \in C} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}) p(o \mid c, a)} \right)$$

The derivation of Equation 2.4 from the definition of mutual information is given in Appendix A.2. It is based on Denzler and Brown (2000) and makes use of the Markov assumption. Equation 2.4 shows that the mutual information for an action can be calculated on the basis of the belief state $p(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})$ and the observation probability distributions $p(O \mid c, a)$. MI does not have any model parameters that have to be trained. Hence, this active vision model does not need any training time other than the time to learn the observation probability distributions $p(O \mid c, a)$.

In Denzler and Brown (2002), an approach for dealing with continuous observations or states is proposed that employs Monte-Carlo sampling from the observation probability distributions. We implemented this approach, since in our task of classifying 3-D objects (described in Section 2.4) the models face continuous observations. As a consequence, the execution time of the model depends on both the number of objects and the number of actions. This leads to an increasing amount of execution time for increasing object subset sizes. We note that this is a consequence of the sampling used, not of the criterion of mutual information itself. See, for example, the research by Denzler, Zobel, and Niemann (2003).

2.3.2 Learning for Entropy Reduction (L-ER)

The second model, learning for entropy reduction (L-ER), adapts an action selection policy to the task (see, e.g., Paletta *et al.*, 1998; Deinzer *et al.*, 2003). For example, we can adapt a policy Π that maps the belief state to an action, $\Pi : p(C \mid \mathbf{o}_i, \mathbf{a}_i) \rightarrow A$. In Paletta *et al.* (1998) the mapping is adapted by means of reinforcement learning

(see, e.g., Sutton and Barto, 1998), with the entropy reduction in the belief state as the reinforcement signal. Instead of reinforcement learning, we employed an evolutionary algorithm (Holland, 1992; Bäck, 1996).

We implemented Π with a feedforward multilayer neural network. The neural network transformed the current belief state into outputs that represented the possible actions. Therefore, the network had n input neurons, where n was the number of objects (classes) under consideration. It had 36 output neurons, i.e., the number of possible actions (see Subsection 2.4.1). The active vision model executed the action corresponding to the output neuron with the highest activation value. The number of hidden neurons was half the number of inputs (classes), i.e., $\lfloor \frac{n}{2} \rfloor$. The hidden and output neurons all had sigmoid activation functions: $a(z) = \tanh(z) = 1 - \frac{2}{1+e^{2z}}$, $a(z) \in \langle -1, 1 \rangle$. The model parameters to be trained for the task were the weights of the neural network. They formed the genome used by the evolutionary algorithm[4] (see Subsection 2.4.4).

The fitness f of policies in the population during evolution was defined as the expected total entropy reduction in the belief state over all t time steps.

$$f(\Pi) = E\left[\sum_{i=1}^{t} \left(H(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}) - H(C \mid \mathbf{o}_i, \mathbf{a}_i)\right)\right] = E\left[H(C) - H(C \mid \mathbf{o}_t, \mathbf{a}_t)\right] \tag{2.5}$$

We approximated the expected total entropy reduction as follows. First a policy Π was executed on all objects in the selected object set. Then we averaged over the measured total entropy reduction.

A key property of L-ER is that its action policy allows very fast action selection at execution time. However, this fast execution time comes at the cost of a long training time. In addition, if we add an object to the object subset, we need to retrain L-ER's action policy to adjust to the new situation.

2.3.3 Ranking (RK)

The third model, ranking (RK), bases its action selection on a ranking of actions made in advance of the model's execution (see, e.g., Schiele and Crowley, 1998; Kröse and Bunschoten, 1999; Arbel and Ferrie, 2001). The model ranks the actions $a \in A$ for each ground truth class $k \in C$ according to how well they disambiguate the different classes. To determine how well an action disambiguates the classes, the model uses the expected entropy reduction, given a uniform belief state. The training of the model consists of assigning values to the actions. The values are the model parameters trained for the task, and they are set according to Equation 2.6. In the equation, the variable k denotes the ground truth class, since the variable c already denotes the different classes in the context of the belief state.

[4] Since the model parameters were trained by evolutionary adaptation, we may also refer to them as *adaptable model parameters*. This is the term that we will use in the chapters 4 to 6, since in those chapters we only use evolutionary algorithms.

$$\text{value}(a \mid k) = E\left[H(C) - H(C \mid o, a) \mid a, k\right] \tag{2.6}$$

$$= H(C) - E[H(C \mid o, a) \mid a, k] \tag{2.7}$$

We approximated the expected entropy in Equation 2.6 by sampling repeatedly an observation $o \in O$ from class k, according to $p(O \mid a, k)$, resulting in observations $o_j(a, k)$, $j = 1, 2, \ldots, m$. Then we used the following approximation.

$$E[H(C \mid o, a) \mid a, k] \approx \frac{1}{m} \sum_{j=1}^{m} H(C \mid a, o_j(a, k)) \tag{2.8}$$

During execution, the model receives an observation, updates the belief state, and selects the best action for the most probable class. It selects the action with the highest value that has not been performed before. Compared to MI, RK has a short execution time, since the ranking of actions resulting from the training procedure allows a fast execution time. Compared to L-ER, it has a short training time, since it calculates only once for every action-class pair how good the action is. Yet, the drawback of RK may be that the advantages concerning execution and training time come at the cost of a lower performance attained on the classification task.

2.3.4 Learning for Classification Performance (L-CP)

The fourth model, learning for classification performance (L-CP), is inspired by the adaptive approach to active vision. The model is rather similar to the model L-ER. They both employ an action policy Π that maps the belief state to an action, $\Pi : p(C \mid \mathbf{o}_i, \mathbf{a}_i) \rightarrow A$. The difference with L-ER is that we use a different fitness function f for this model:

$$f(\Pi) = \frac{g}{n}, \tag{2.9}$$

where g is the number of correctly classified objects, in a trial in which the model has to classify n objects ($n \geq g$). Since the model is so similar to L-ER, L-CP is only a small step in the direction of the adaptive approach. The reason that we still state that this model belongs to the adaptive approach, is that it directly uses the classification performance for learning the action strategy. We implemented the action policy Π with the same type of feedforward multilayer neural network as the one used by L-ER. Both methods train the same model parameters, i.e., the neural network weights. As L-ER, L-CP has a long training time and a short execution time.

2.3.5 Random Actions (RA)

The fifth model, random actions (RA), selects actions at random. It is commonly used as a benchmark model in experiments with active vision models. The difference in performance between this model and the more elaborate models discussed above forms an indication on how useful it is to use their action strategies. We note that one should not think of this model as randomly selecting a class: although the model takes random actions, it uses the same estimates of the observation probability distributions and the same belief state update rule as the other active vision models. It may be clear that RA does not have model parameters that need to be trained.

2.3.6 Passive Vision Model (PA)

The sixth model, the passive vision model (PA), receives one observation and immediately classifies the object with the help of the observation probability distribution. It selects the class that has the highest probability to generate the observation o, given its randomly selected a. The model provides a bottom performance for all active vision models described above. As RA, PA does not have model parameters that need to be trained.

2.4 Experimental Setup

In this section, we describe the experimental setup. We first explain the task to which the models are applied. Then we discuss all experimental conditions and indicate how the results are obtained for each condition. Subsequently, we explain the evolutionary algorithm that is used to train the models L-ER and L-CP.

2.4.1 Turn-table Task

Most of the probabilistic active vision models have been introduced in the context of a classification task of 3-D objects (cf. Paletta *et al.*, 1998; Borotschnig *et al.*, 2000; Arbel and Ferrie, 2001; Denzler and Brown, 2002). Therefore, we also employ such a task for our empirical comparison of the active vision models. For the experiments, we used the **A**msterdam **L**ibrary of **O**bject **I**mages (ALOI) data set (Geusebroek, Burghouts, and Smeulders, 2005). The data set contains 1000 objects. Each object has been placed on a *turn-table* and has been photographed from 72 angles that are each 5° apart. In our experiments, we used the set of gray-value images with the smallest size (192×144). Figure 2.1 shows an example subset of 25 objects.

The task for the active vision model is to classify a 3-D object, by selecting the right viewing angles. The selection of a viewing angle corresponds to turning the turn-table. Figure 2.2 illustrates one time step in the classification task. The active vision model has to determine an action on the basis of its current belief state (left box). The actions that the model can take correspond to the angles from which it can view the object. The selected action results in an observation (top right box). An observation is a vector containing the values of the first q principal

Figure 2.1: Example set of objects ($n = 25$).

components of the image taken from the selected viewing angle (cf. Paletta *et al.*, 1998; Borotschnig *et al.*, 2000; Denzler and Brown, 2002). Figure 2.3 is a visualisation of the first five principal components of the object set in Figure 2.1. Since different classes could have led to the observation, the active vision model assigns probabilities to the different classes on the basis of the observation probability distributions (bottom right box). On the basis of these probabilities, the belief state is updated (left box). As in Denzler and Brown (2002), the active vision model classifies the object, as soon as its confidence in one of the classes is higher than 90% ($\max(p(c \mid \mathbf{o}_i, \mathbf{a}_i)) \geq 0.90$) or the number of observations is equal to 10. In these cases, the class k with highest probability is selected: $k = \text{argmax}_c p(c \mid \mathbf{o}_i, \mathbf{a}_i)$, else the model continues selecting viewing angles and updating its belief state.

2.4.2 Experimental Conditions

In total, we performed experiments under 16 different conditions. A main difference between the various experimental conditions is given by the fact whether the model can determine the first viewing angle or not.

If the model can determine the first viewing angle, it starts with a uniform belief state. If it cannot determine the first angle, the run starts with an observation from a random angle that is unknown to the active vision model. Therefore, the model has to follow a different strategy for updating the belief state for the first observation.

In our experiments, we examined two different strategies: a *conservative strategy* and a *confident strategy*. Both strategies take the observation o and determine for each action-class pair the probability that it leads to the observation. The conservative strategy then excludes the classes that have zero probability of leading to the observation, assigning equal probabilities to the remaining classes. The confident strategy sums the probabilities over all classes per action. It assumes that the action

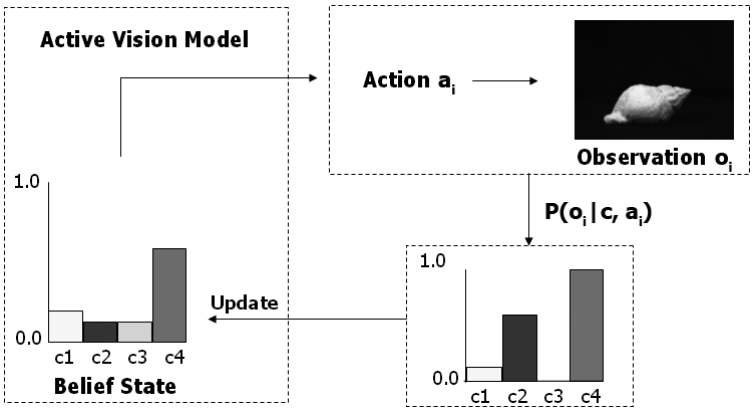


Figure 2.2: Illustration of one time step in the turn-table task. The active vision model selects an action on the basis of its current belief state (left box). This action corresponds to the angle from which the object is viewed, and results in an observation (top right box). The model assigns probabilities to the different classes on the basis of the observation probability distributions (bottom right box). On the basis of these probabilities, the belief state is updated (left box).

Figure 2.3: Visualisation of the first five principal components of the object set shown in Figure 2.1.

taken is the one with the highest sum of probabilities and updates the belief state with the help of o and the most probable action.

In addition to varying whether the model can select the first viewing angle and what update strategy it follows for the first observation, we also varied two task parameters[5]: the number of objects that have to be classified (n), and the number of principal components (q). This leads to four factors of variation in the experimental conditions.

1. Selection of first viewing angle
2. Confident or conservative strategy
3. Number of objects n
4. Number of principal components q

[5]Parameters pertain to *quantities*, therefore we do not speak of task parameters in the case of selecting the first view angle and choosing the update strategy of the first observation.

Table 2.1 summarises the different experimental conditions. We note that not all possible conditions are studied, since this would be intractable. The 16 experimental conditions are subdivided into four sets (A, B, C, and D). In sets A, C, and D, we use $q = 1$ principal component, in set B we use $q = 2$ principal components. The number of principal components does not exceed 2, since adding more components leads to a ceiling effect (see Section 2.5). Furthermore, for each set we vary the number of objects n with steps of 25 from 25 to 100 in order to test the active vision models on different levels of difficulty. We stop at 100, since adding objects does not only make the task more difficult, it also increases the training / execution time of the active vision models. The number of runs is explained in Subsection 2.4.3.

Table 2.1: The 16 experimental conditions used for comparing the six vision models. The conditions are subdivided into four sets (A, B, C, and D). q stands for the number of principal components, n for the object subset size. For the experiments with a random first view angle, we employed a conservative or a confident strategy to cope with the first observation. The meaning of the number of runs is explained in Subsection 2.4.3.

		Model Determines First View Angle	
Set A	$q = 1$	$n = 25, 50, 75, 100$	50 runs
Set B	$q = 2$	$n = 25, 50, 75, 100$	30 runs
		Random First View Angle	
Set C	$q = 1$	$n = 25, 50, 75, 100$ - conservative strategy	30 runs
Set D	$q = 1$	$n = 25, 50, 75, 100$ - confident strategy	30 runs

Before starting the experiments under all experimental conditions, we manually tuned all parameters that are not subject to training. We have chosen to tune the relevant model parameters and training parameters (see Subsection 2.4.4) on a set of 250 objects from the ALOI data set, henceforth referred to as the tuning set. Then, we used the tuned model parameters and training parameters to perform our experiments under the 16 experimental conditions on the 750 objects that were not used for tuning, i.e., the testing set.

2.4.3 Experimental Runs

The results of the 16 experimental conditions have been obtained by performing multiple experimental runs. We regarded the experiments of set A, i.e., with one principal component ($q = 1$) and in which the models determine the first viewing angle, as our standard experiments. To obtain the most reliable results for these experiments, we performed 50 different experimental runs. For the other experiments we relied on 30 different experimental runs (see Table 2.1). Each experimental run consisted of four phases. They are enumerated below, and thereafter discussed in more detail.

1. Object subset selection and preparation
2. Estimation of the observation probabilities
3. Training of the vision models

4. Testing of the vision models

Phase 1: Object Subset Selection and Preparation

In phase 1 we randomly selected an object subset of size n from the ALOI data set. We loaded the corresponding images and resized them to one fourth of their size with bicubic resampling (using the MATLAB© implementation). Following the experimental setup in Schiele and Crowley (1997), the images were divided into a training set (images with angles divisible by 5°, excluding those divisible by 10°) and a test set (remaining images). *Simple PCA* (Partridge and Calvo, 1998) was used to determine the principal components in the training images.

Phase 2: Estimation of the Observation Probabilities

In phase 2 we estimated the observation probabilities $p(O|A, C)$ that are necessary for the belief state updates of all the models. For simplicity, we assumed, as in Kröse and Bunschoten (1999), Borotschnig *et al.* (2000), and Denzler and Brown (2002), that the observations O for a given action a and class c were Gaussian distributed in the space spanned by the principal components: $p(O|a, c) \sim \mathcal{N}(\mu, \Sigma)$, with μ and Σ dependent on a and c. The covariance matrix Σ was assumed to be diagonal. The distribution parameters μ and Σ were estimated for each combination of a and c with the help of the training set. The difficulty in determining the observation distributions was that the ALOI data set only contains one image per class and per angle. It is not possible to estimate an observation probability distribution on the basis of one observation. Therefore, we increased the number of images used for the estimation by perturbing the training images. This perturbation took into account that the photos in the test set belong to the angles in between those of the training set. In particular, to sample an observation from class c and angle $d°$, we *morphed* the image with either the image of angle $d + 10°$ or the image of angle $d - 10°$. The morphed image vector v_{morph} was determined as follows.

$$v_{\text{morph}} = \begin{cases} (1 - |m|)v_d + |m|v_{d+10} & \text{, if } m > 0 \\ (1 - |m|)v_d + |m|v_{d-10} & \text{, if } m \leq 0 \end{cases} \tag{2.10}$$

The result of morphing depended on the random number $m \in [-0.5, 0.5]$. Figure 2.4 shows three real images and two morphed images of a sneaker viewed from angles $d = 20$, 30, and 40. The images indicated by $m = -0.5$ and $m = 0.5$ are morphed images of the image with $d = 30$.

Phase 3: Training of the Vision Models

In phase 3 we trained the active vision models L-ER, RK, and L-CP on the training angles. The models MI, RA, and PA do not have a training procedure. We have fully described the training procedure of RK in Subsection 2.3.3. We trained L-ER and L-CP with an evolutionary algorithm, which is explained further in Subsection 2.4.4.

Figure 2.4: Morphing images for estimation of the observation probability distributions. The left image is taken from angle $d = 20$, the middle image from $d = 30$, and the right image from $d = 40$. The images indicated with $m = -0.5$ and $m = 0.5$ are morphed images.

Phase 4: Testing of the Vision Models

In phase 4 we tested the active vision models on the test angles. We provided each model with the same test objects and, in the case that the model did not determine the first viewing angle, with the same first test viewing angles.

2.4.4 Evolutionary Algorithm

In this subsection, we discuss the λ, μ-evolutionary algorithm (Bäck, 1996) employed to train the active vision models L-ER and L-CP. As explained above, we tuned the training parameters, such as the number of policies λ and the number of selected policies μ, on the tuning set of objects. The influence of different training parameter values on the results is discussed further in Subsection 2.7.4. For the time being, we just explain the training parameters and mention their tuned values.

We started evolution by randomly initialising λ different policies, $\lambda = 20$. Each policy $\Pi : p(C \mid \mathbf{o}_i, \mathbf{a}_i) \rightarrow A$ was implemented by a fully connected multilayer feedforward neural network, with weights in the range [-1, 1]. The genome of an individual policy directly represents the weights as a vector of double values in the aforementioned range.

Training observations were generated according to the learned Gaussian distributions that represent $p(O|a, c)$, but we multiplied the standard deviations of these distributions by a constant factor β in order to obtain interesting training cases that require actions. Figure 2.5 illustrates why we employed this factor β. The solid and dotted line illustrate the observation probability distributions of two different objects, projected onto the first principal component. Clearly, if we sample according to these probability distributions, each observation can be classified correctly without the need for further actions. However, if the factor β is employed, the generated observations are more interesting. For example, the model may encounter observations that fall in between the probability distributions of the two different objects. The dashed line in Figure 2.5 illustrates the way in which observations are sampled for training. We set β to 4, on the basis of preliminary experiments on the tuning set of objects.

To evaluate a policy, we applied it n times to the training set, so that it was evaluated once for each object. For L-CP we defined the fitness of a policy as the

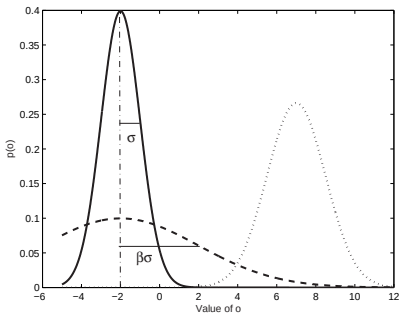


Figure 2.5: Illustration of the use of β. The solid and dotted line illustrate the observation probability distributions for the same action of two different objects, of which the images are projected on the first principal component. The dashed line illustrates the way in which we sample observations for training if the model has to classify the object corresponding to the solid line, with $\beta = 4$.

proportion of correct classifications. For L-ER we defined the fitness of a policy as the mean entropy-loss in the belief state. After all policies had been assigned a fitness, the best $\mu = 5$ policies were selected to form the next generation of policies. Per selected policy, 4 offspring were created by copying its genome 4 times. Then mutation was applied to the genomes of the offspring, where every gene (weight of the neural network) had a probability of p_{mut} of being mutated. In our experiments, we set p_{mut} to $\frac{1}{25}$. We mutated a gene by assigning it a random number from the interval $[-1, 1]$. This process of fitness evaluation, selection, and procreation, continued for $2n$ generations. Preliminary experiments on the tuning set had shown that this number of generations allowed convergence of the evolutionary algorithm. The best policy of the last generation was returned as the trained policy. Since the outcome of training depends on the initial population, we performed three evolutions and selected the policy that obtained the highest fitness. We evaluated this selected policy on the test set.

2.5 Results

As explained in Subsection 2.4.2, we performed experiments under 16 different conditions, subdivided into four sets A, B, C, and D. In this section, we first show the results of the experiments in which the first view angle is determined by the models (sets A and B). Then we present the results of the experiments in which the first view angle is determined at random and is unknown to the models (sets C and D).

2.5.1 Model Determining First View Angle

Below, we first show the results for the experiments with one principal component (set A), and then for two principal components (set B). In each case, experiments were performed for $n = 25, 50, 75,$ and 100.

Set A: One Principal Component

Table 2.2 shows the results of all six vision models for the experiments with one principal component. Table 2.2 consists of three subtables, which we describe below.

Table 2.2: Results of experiments in set A ($q = 1$). **Top table:** Average performance and standard error of the mean per vision model, based on 50 experimental runs. The best average performances are typeset in bold. **Middle table:** Number of actions performed per vision model, based on 50 experimental runs. The first number in each cell is the average number of actions performed per run. The second number is the average of the maximal number of actions performed for one object in an object subset. **Bottom table:** Statistically significant results for the experiments with $n = 25, 50, 75,$ and 100 ($p < 0.05$). If a subset size is mentioned in a cell, then the active vision model of the cell's row significantly outperforms the model of the cell's column.

n	MI	L-ER	RK	L-CP	RA	PA
25	84.6 (1.3)	86.1 (1.1)	81.8 (1.3)	**87.0 (1.0)**	74.4 (0.1)	68.6 (0.2)
50	**73.6 (1.2)**	71.0 (1.0)	71.1 (1.0)	72.1 (1.0)	64.0 (1.0)	47.9 (1.1)
75	63.2 (0.9)	61.4 (0.9)	62.2 (0.9)	**64.7 (1.0)**	58.4 (0.9)	34.2 (1.0)
100	56.8 (0.9)	54.3 (0.9)	55.6 (0.9)	**62.2 (1.3)**	58.5 (0.6)	27.9 (0.9)

n	MI	L-ER	RK	L-CP	RA	PA
25	1.06 / 1.82	1.04 / 1.68	1.04 / 1.64	1.05 / 1.74	1.19 / 2.26	1 / 1
50	1.09 / 2.00	1.08 / 2.08	1.08 / 2.08	1.08 / 2.00	1.30 / 2.67	1 / 1
75	1.12 / 2.14	1.11 / 2.22	1.17 / 2.44	1.18 / 2.38	1.15 / 3.06	1 / 1
100	1.15 / 2.10	1.16 / 2.40	1.22 / 2.74	1.35 / 2.54	1.52 / 3.20	1 / 1

	MI	L-ER	RK	L-CP	RA	PA
MI					25, 50, 75	25, 50, 75, 100
L-ER			25		25, 50, 75	25, 50, 75, 100
RK					25, 50, 75	25, 50, 75, 100
L-CP	100	75, 100	25, 100		25, 50, 75, 100	25, 50, 75, 100
RA		100	100			25, 50, 75, 100
PA						

The top table shows the mean performance and standard error of the mean. The results of each model are shown in a column, while rows represent the experiments for a specific number of objects. We type-set the results of the model that has the best average performance in bold. We may make four observations about these results. First, the table shows that for 25, 50, and 75 objects, all methods have a higher mean performance than the random active vision strategy, confirming the results from the literature (Paletta *et al.*, 1998; Borotschnig *et al.*, 2000; Denzler and Brown, 2002). However, it also shows that the random strategy deteriorates less

with a growing object subset. For 100 objects, the random strategy performs better than all other models, except L-CP. Second, model RK performs worse than models MI, L-ER, and L-CP on the smallest subset, but performs better than model L-ER on the bigger subsets with an increasing difference. Third, the active vision model L-CP performs better than model L-ER on all subset sizes with an increasing difference. It attains the highest performance on three of the four subsets, and is the only active vision model that outperforms the random active vision strategy for 100 objects. Fourth, MI and L-CP always perform better than RK.

The middle table shows the number of actions that the different active vision models take to classify an object. It shows per subset size and per vision model, how many actions it takes on average to classify an object (first number in each cell). We also recorded for each experimental run the maximum number of actions taken by each active vision model. The table shows what the maximum number of actions is that a model takes, averaged over all experimental runs (second number in each cell). We observe from the table that on average RA performs the most actions, with as only exception the subset size of 75 objects. The higher number of actions might explain why RA outperforms most other models on the subset size of 100. Between the other active vision models, there do not seem to be big differences in the number of actions that they take. The average number of actions for all models and all subsets is close to 1, indicating that the first action often already leads to the classification of the object under evaluation.

The bottom table shows what performance differences are statistically significant. Since we do not want to make any assumptions regarding the probability distributions of the experimental results, statistical significance is determined with a randomisation test (Cohen, 1995), with $p < 0.05$. In each cell of the table we include the object subset sizes for which the row's model is significantly better than the column's model. For example, L-ER significantly outperforms RK for a subset size of 25. Therefore, we include 25 in the cell with row L-ER and column RK.

Set B: Two Principal Components

The task becomes easier if we add a principal component to the observation vector. Table 2.3 shows the outcome of the experiments with two principal components. The top table shows that the objects in the ALOI data set are rather dissimilar, since the addition of a principal component causes a ceiling effect for the active vision model performances. The performance differences of the passive vision model PA with the other models is smaller than in the experiments with one principal component. As the subset size grows, the active vision models increasingly outperform RA and PA. The middle table shows the average number of actions and the average maximum number of actions of all active vision models for the task with two principal components. As to be expected, the number of actions in Table 2.3 are all closer to 1 than those in Table 2.2. The bottom table shows the statistical significance of the performance differences.

Table 2.3: Results of experiments in set B ($q = 2$). **Top table:** Average performance and standard error of the mean per vision model, based on 30 experimental runs. **Middle table:** Number of actions performed per vision model, based on 30 experimental runs. The first number in each cell is the average number of actions performed per run. The second number is the average of the maximal number of actions performed for one object in an object subset. **Bottom table:** Statistically significant results for the experiments with $n = 25, 50, 75$, and 100 ($p < 0.05$).

n	MI	L-ER	RK	L-CP	RA	PA
25	99.6 (0.3)	**99.7 (0.3)**	99.3 (0.3)	**99.7 (0.3)**	95.1 (0.9)	97.9 (0.5)
50	**98.5 (0.4)**	98.3 (0.4)	97.7 (0.4)	**98.5 (0.3)**	92.2 (0.6)	92.2 (0.8)
75	97.3 (0.4)	**97.7 (0.3)**	97.3 (0.4)	97.6 (0.3)	91.2 (0.6)	88.5 (0.7)
100	95.8 (0.6)	**96.2 (0.5)**	95.7 (0.6)	96.1 (0.5)	86.0 (0.5)	86.0 (0.7)

n	MI	L-ER	RK	L-CP	RA	PA
25	1.00 / 1.03	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00	1.01 / 1.13	1 / 1
50	1.00 / 1.20	1.00 / 1.10	1.00 / 1.13	1.00 / 1.10	1.02 / 1.60	1 / 1
75	1.01 / 1.47	1.01 / 1.30	1.01 / 1.33	1.00 / 1.23	1.04 / 1.93	1 / 1
100	1.01 / 1.50	1.00 / 1.37	1.00 / 1.40	1.00 / 1.37	1.04 / 2.03	1 / 1

	MI	L-ER	RK	L-CP	RA	PA
MI					25, 50, 75, 100	25, 50, 75, 100
L-ER					25, 50, 75, 100	25, 50, 75, 100
RK					25, 50, 75, 100	25, 50, 75, 100
L-CP					25, 50, 75, 100	25, 50, 75, 100
RA						75
PA					25	

2.5.2 Random First View Angle

In this subsection, we show the results of the experiments in which the first view angle is selected at random. We do so for a conservative strategy (set C) and for a confident strategy (set D). Again 30 experimental runs were performed for $n = 25, 50, 75$, and 100.

Set C: Conservative Strategy

Table 2.4 shows the results for the task in which the first view is selected at random, where the model employs the conservative strategy to handle the first observation. We see in the top table that the performance of PA is much lower than in Table 2.2. The reason for this is that it is much harder to classify an object on the basis of one observation, if the angle is unknown. Many of the other performances in the table are higher than those in Table 2.2. The reason for this becomes evident, when investigating the number of actions executed by the active vision models, shown in the middle table. The models take on average roughly one extra action when they do not determine the first angle. This extra action is the random, unknown angle. Apparently, the first observation leads to the exclusion of some objects, but

does not often lead to an immediate classification. With one observation more, it is not surprising that the performances are higher than for set A in Subsection 2.5.1. The only active vision model that does not perform better, is L-CP. Concerning the comparison of the active vision models, the main observation is that MI performs better than the other models on all three subset sizes. The bottom table shows the statistical significance of the performance differences between the various active vision models.

Table 2.4: Results of experiments in set C (conservative strategy). **Top table:** Average performance and standard error of the mean per vision model, based on 30 experimental runs. **Middle table:** Number of actions performed per vision model, based on 30 experimental runs. The first number in each cell is the average number of actions performed per run. The second number is the average of the maximal number of actions performed for one object in an object subset. **Bottom table:** Statistically significant results for the experiments with one principal component and random first angle for $n = 25, 50, 75,$ and 100 ($p < 0.05$).

n	MI	L-ER	RK	L-CP	RA	PA
25	**89.3 (1.1)**	87.0 (1.2)	80.2 (1.5)	87.0 (1.2)	76.0 (1.0)	21.8 (1.4)
50	**75.0 (1.2)**	73.5 (1.3)	72.5 (1.4)	72.0 (1.2)	65.0 (0.7)	12.1 (1.3)
75	**66.9 (1.1)**	63.0 (1.3)	66.2 (0.9)	63.8 (1.2)	60.3 (0.5)	9.3 (0.9)
100	60.5 (5.3)	56.9 (5.6)	**63.5 (6.0)**	63.0 (8.6)	58.7 (2.2)	8.6 (5.0)

n	MI	L-ER	RK	L-CP	RA	PA
25	2.04 / 3.03	2.04 / 2.73	2.09 / 2.90	2.04 / 2.60	2.16 / 3.87	1 / 1
50	2.08 / 3.10	2.07 / 3.10	2.16 / 3.30	2.08 / 3.17	2.29 / 4.37	1 / 1
75	2.13 / 3.27	2.12 / 3.20	2.27 / 3.57	2.13 / 3.07	2.41 / 4.83	1 / 1
100	2.15 / 3.13	2.15 / 3.33	2.34 / 3.83	2.29 / 3.53	2.50 / 5.03	1 / 1

	MI	L-ER	RK	L-CP	RA	PA
MI		75, 100	25		25, 50, 75	25, 50, 75, 100
L-ER			25		25, 50, 75	25, 50, 75, 100
RK		75, 100			25, 50, 75, 100	25, 50, 75, 100
L-CP		100	25		25, 50, 75, 100	25, 50, 75, 100
RA						25, 50, 75, 100
PA						

Set D: Confident Strategy

We now turn to the results of the experiments with the confident strategy. Table 2.5 shows the results of these experiments. In the top table, we can see that the performances obtained are lower than for the conservative strategy. Comparing the middle tables of the conservative strategy (Table 2.4) and the confident strategy (Table 2.5) reveals that, on average, the active vision models take fewer actions for the confident strategy than for the conservative strategy. Assuming that the random action corresponds to the most probable action results in faster, but faultier classifications. Table 2.5 shows that L-CP and MI perform best for this strategy. The bottom table again shows the performance differences that were statistically significant.

Table 2.5: Results of experiments in set D (confident strategy). **Top table:** Average performance and standard error of the mean per vision model, based on 30 experimental runs. **Middle table:** Number of actions performed per vision model, based on 30 experimental runs. The first number in each cell is the average number of actions performed per run. The second number is the average of the maximal number of actions performed for one object in an object subset. **Bottom table:** Statistically significant results for the experiments with one principal component and random first angle for $n = 25, 50, 75,$ and 100 ($p < 0.05$).

n	MI	L-ER	RK	L-CP	RA	PA
25	49.9 (1.2)	43.1 (1.3)	47.8 (1.1)	**50.2 (1.4)**	39.9 (0.8)	23.6 (1.8)
50	**51.3 (0.8)**	46.3 (1.2)	47.3 (0.9)	51.2 (0.9)	37.0 (0.5)	13.0 (1.2)
75	47.6 (0.8)	46.4 (1.0)	41.9 (0.7)	**48.1 (0.8)**	32.3 (0.6)	7.2 (0.7)
100	44.5 (3.2)	44.2 (3.7)	39.2 (3.1)	**45.7 (3.6)**	30.1 (2.3)	6.8 (4.1)

n	MI	L-ER	RK	L-CP	RA	PA
25	1.57 / 3.03	1.57 / 3.27	1.57 / 3.17	1.59 / 3.23	1.57 / 3.23	1 / 1
50	1.86 / 3.23	1.88 / 3.67	1.86 / 3.30	1.89 / 3.60	1.87 / 3.73	1 / 1
75	1.95 / 3.27	1.97 / 3.62	1.96 / 3.62	1.97 / 3.96	1.97 / 4.00	1 / 1
100	1.98 / 3.67	2.00 / 3.87	1.99 / 3.73	2.00 / 3.93	2.02 / 4.23	1 / 1

	MI	L-ER	RK	L-CP	RA	PA
MI		25, 50	50, 75, 100		25, 50, 75, 100	25, 50, 75, 100
L-ER			75, 100		25, 50, 75, 100	25, 50, 75, 100
RK		25			25, 50, 75, 100	25, 50, 75, 100
L-CP		25, 50	50, 75, 100		25, 50, 75, 100	25,50, 75, 100
RA						25, 50, 75, 100
PA						

2.6 Analysis

In this section, we analyse the empirical results. We start by providing a summary of the results of the 16 experiments. Then, we attempt to explain the performance differences between the three probabilistic models MI, L-ER, and RK. Subsequently, we analyse the performance differences between the three probabilistic models, MI, L-ER, and RK on the one hand, and the adaptive model L-CP on the other hand. We do not analyse the performance differences between the probabilistic / adaptive models with the random active vision model RA or the passive vision model PA, since this would be trivial.

2.6.1 Summary of Results

Table 2.6 is a summary of the results of the 16 experiments concerning the performance differences between the models (Section 2.5). Each cell contains four numbers. The top two numbers (x - y) represent the number of times that the model of the row had a higher (x) or a lower (y) average performance than the model of the column. The bottom two numbers (v - w) represent the number of times that the model of the row was significantly better (v) or significantly worse (w) than the model of the column. For example, the third cell of the top row indicates that MI outperformed RK 14 times (of which 4 times significantly), while RK outperformed MI 1 time (not significantly).

Table 2.6: Summary of performance differences of all experiments. For the precise interpretation, we refer to Subsection 2.6.1.

	MI	L-ER	RK	L-CP	RA	PA
MI		12 - 4 4 - 0	14 - 1 4 - 0	5 - 10 0 - 1	15 - 1 14 - 0	16 - 0 16 - 0
L-ER	4 - 12 0 - 4		9 - 7 4 - 3	3 - 11 0 - 5	14 - 2 14 - 1	16 - 0 16 - 0
RK	1 - 14 0 - 4	7 - 9 3 - 4		3 -13 0 - 6	15 - 1 15 - 1	16 - 0 16 - 0
L-CP	10 - 5 1 - 0	11 - 3 5 - 0	13 - 3 6 - 0		16 - 0 16 - 0	16 - 0 16 - 0
RA	1 - 15 0 - 14	2 - 14 1 - 14	1 - 15 1 - 15	0 - 16 0 - 16		13 - 1 13 - 1
PA	0 - 16 0 - 16	0 - 16 0 - 16	0 - 16 0 - 16	0 - 16 0 - 16	1 - 13 1 - 13	

Table 2.6 shows an overview of the performance differences between the six vision models on the 16 experiments. Table 2.7 provides an impression of which models perform better than others. A '$>$'-sign in a cell means that the model of the row outperformed the model of the column, and a '$<$'-sign the opposite. A '$\approx$'-sign means that the experimental results did not provide a clear result. In the

next subsections, we analyse the causes of the performance differences between the different active vision models.

Table 2.7: Relative performances of the six vision models. '$>$' in a cell means that the model of the row outperformed the model of the column, and '$<$' the opposite. '$\approx$' means that the experimental results did not provide a clear result.

	MI	L-ER	RK	L-CP	RA	PA
MI		$>$	$>$	$\approx$	$>$	$>$
L-ER	$<$		$\approx$	$<$	$>$	$>$
RK	$<$	$\approx$		$<$	$>$	$>$
L-CP	$\approx$	$>$	$>$		$>$	$>$
RA	$<$	$<$	$<$	$<$		$>$
PA	$<$	$<$	$<$	$<$	$<$	

2.6.2 Differences between MI, L-ER, and RK

We start our analysis with the three probabilistic models MI, L-ER, and RK. We first compare the models MI and L-ER, and then include RK into the comparison.

MI and L-ER

There are three important differences between MI and L-ER. The first important difference concerns the manner in which the models attempt to reduce the uncertainty in the belief state. The active vision model MI selects an action $a_i = \text{argmax}_a I(C; O \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)$. In order to compare MI adequately with L-ER, we reformulate MI's action selection (see Equation 2.3) as follows.

$$a_i = \text{argmax}_a E\left[H(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}) - H(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a, o)\right] \tag{2.11}$$

$$= \text{argmax}_a \sum_{c \in C} \sum_{o \in O} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) p(o \mid c, a) \log \left(\frac{p(o \mid c, a)}{\sum_{c \in C} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}) p(o \mid c, a)} \right) \tag{2.12}$$

Here o is the observation that results from action a. MI maximises the expected entropy loss in the belief state, given the belief state and the probability distribution of observations O for an action a.

In contrast, L-ER selects actions $\mathbf{a}_t^*$ according to the following formula.

$$\mathbf{a}_t^* = \text{argmax}_{\mathbf{a}_t} E[H(C) - H(C \mid \mathbf{o}_t, \mathbf{a}_t)] \tag{2.13}$$

In the formula t is the time step at which a class is assigned to the object under consideration. Comparing Equation 2.13 with Equation 2.11 makes the first important difference salient: L-ER maximises the entropy reduction over multiple time steps, while MI maximises the entropy reduction over one time step. As a consequence, an advantage of L-ER is that it can perform non-greedy action selection while MI always performs greedy action selection. We note that greediness is not a general property of active vision models based on the criterium of mutual information. It is possible to remodel the active vision process, so that the model can select actions that lead to maximal uncertainty reduction over multiple time steps, as in Deutsch *et al.* (2004).

The second important difference between MI and L-ER concerns the difference between calculating and learning good actions. Where MI calculates the expected entropy reduction at every time step, L-ER learns an action mapping on the basis of experience. During training it performs an action a, which leads to a specific observation o, and to an entropy loss in the belief state:

$$H(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}) - H(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a, o). \tag{2.14}$$

While MI directly uses the distribution $p(O \mid c, a)$ to calculate the expected entropy loss (see Equation 2.12), the action mapping of L-ER averages over the observations received during training. These observations come from the distribution $p(O \mid c, a)$. Therefore, with an unlimited amount of training time, L-ER should arrive at the same actions as MI. However, in our implementation, L-ER has a limited amount of training time. Therefore, it might take actions that do not lead to a maximal expected entropy reduction.

The third important difference between MI and L-ER concerns the generalisation over the belief state. L-ER employs a neural network for implementing Π. The neural network generalises over the input space $p(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})$. Therefore, similar belief states are mapped to similar actions, even if using $E[H(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}) - H(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a, o)]$ suggested a different action. Since MI calculates the expected entropy reduction for each action, it can more accurately map the belief state to different actions than L-ER.

Table 2.6 shows that MI generally performs better than L-ER. In the current 3-D object classification task, L-ER cannot exploit its advantage of learning non-greedy action strategies. The main reason for this is that the active vision models can change the view angle to any other angle at any time step. Therefore, the active vision models do not need to employ a sequence of non-greedy actions to arrive at a discriminative view angle. Furthermore, the experimental results show that classification can often already be performed with one or two observations. This does not leave much room for employing non-greedy action selection. We believe that the generalisation of the neural network over the input space is the main reason that MI outperforms L-ER on most experimental settings. One could argue that the factor $\beta = 4$ is the reason for L-ER's worse performance, since it prevents L-ER from learning the real observation probability distribution (which would be learned with $\beta = 1$). However, experiments on the tuning set showed that setting β to 4 improved performance.

RK

The differences between MI and L-ER are also important with respect to RK. Model RK uses Equation 2.6 and 2.8 to estimate $E[H(C) - H(C|o, a)|a, k]$. This estimate only considers the mode of the belief state. As a consequence, the amount of resulting input states is equal to the number of classes, $|C|$. This is a low number, implying a coarse generalisation over the input space. This coarse generalisation can lead to suboptimal actions. In addition, RK's estimate does not take into account past observations and actions. Finally, its action selection is also greedy, as that of MI.

The reason that MI outperforms RK is most probably that the latter makes such a coarse generalisation over the input space. We expect that the neglect of information from the past does not play a large role in the task, since it involves a rather low number of actions.

L-ER does not clearly outperform RK in our experiments. This surprised us, because of three reasons: (1) RK makes a rougher generalisation over the belief state, (2) RK neglects information from past observations and actions, and (3) RK uses greedy action selection. The low number of necessary actions most probably removes the effects of the last two reasons. The first reason might be the cause that L-ER outperforms RK twice more than the other way around.

2.6.3 Differences of MI, L-ER, and RK with L-CP

We first compare L-ER with L-CP, since the methods are rather similar. Then, we also compare MI and RK with L-CP.

The only difference between L-ER and L-CP is the fitness function. L-ER is trained on the basis of the entropy reduction in the belief state, while L-CP is trained on the basis of classification performance. In other words, L-ER's action policy is optimised so that the model is as sure as possible about its classification, while L-CP's action policy is optimised so that the model classifies the object correctly.

Intuitively, we may expect that L-CP outperforms L-ER, because being sure about something does not always correspond to being right. In a practical experimental setting, belief state updates on the basis of the estimated probability distributions might not always be correct. Therefore, some belief states might be misleading, resulting in suboptimal actions and possibly wrong classifications. L-ER (and the other two probabilistic models) do not have a means for recovering from such a belief state. L-CP *can* recover from such a misleading belief state, e.g., by selecting an action that does not lead to the maximal entropy reduction but that will lead to correctly classifying the object. We believe that this is the reason why L-CP outperforms L-ER.

The difference between MI and L-CP is more complicated. MI outperforms L-CP five times, while L-CP outperforms MI ten times. A first view would thus lead to the conclusion that L-CP is better than MI. However, there are two reasons why we believe that the experiments are not conclusive on the relation between MI and L-CP. First, the performance differences are almost never statistically significant. Second, the performance difference between the models seems to depend on the

experimental condition. L-CP outperforms MI when the model determines the first action (set A and B), but MI outperforms L-CP on most of the experiments in which the first angle is randomly chosen and a conservative strategy is followed (set C).

RK performs worse than L-CP. Since it performs almost as good as L-ER, the reason for this is probably the same as the one that explains the performance between L-ER and L-CP: RK also attempts to reduce the entropy of the belief state, while this does not always need to lead to a correct classification.

2.7 Discussion

In this section we discuss the limitation of our experiments and the insights provided by the experimental results. These insights concern both the differences between the probabilistic approach and the adaptive approach, and the differences between models of the probabilistic approach. Finally, we discuss the influence of parameter choices for L-CP and L-ER on the results.

2.7.1 Main Limitation

The main limitation of our comparison of the different active vision models is that we compared the active vision models on a view-based 3-D object classification task only. This type of task, although widely used in the literature (e.g., Paletta *et al.*, 1998; Denzler and Brown, 2002), has specific properties that may not be present in many real-world problems. For example, in our task, actions represent absolute angles. As a result, any angle view can be accessed at any time step. In real-world problems, this might not be the case: an action might represent a shift in the angle and larger shifts may require multiple actions. Such a different setup may necessitate changes to the models and may generate different results. For example, it may be wiser to employ another belief state update (e.g., a Naive Bayesian one as in Paletta *et al.*, 1998) or include the pose as a variable to be estimated, and methods that are able to find non-greedy action strategies would perhaps be at an advantage (e.g., L-ER and L-CP). In addition, in real-world problems there might be many more possible actions than just changing the angle from which the model views an object. Additional actions, and a continuous instead of discrete action space might also change the problem significantly, and thus influence the performance differences between the various active vision models.

2.7.2 Probabilistic vs. Adaptive Approach

Even though our comparison is limited, the experimental results give some insight into the differences between probabilistic and adaptive active vision models. One might expect an adaptive model to perform worse on the turn-table task, since the action strategy is not well formalised in terms of reducing entropy. However, the results show that L-CP is not at a disadvantage. The model L-CP has a high performance in most experiments compared to the other active vision models, especially in the experiments where the model can select the first viewing angle. Importantly,

it outperforms L-ER in most experiments, while in general not performing more actions. One could say: *"Of course* L-CP *outperforms* L-ER*; your testing measure is equal to the training measure of* L-CP*."* However, this remark disregards that the final goal of using entropy reduction in L-ER is also to achieve a good classification performance. The results show that in our experiments, the classification performance itself is a better measure than the entropy reduction for training the action selection.

The model L-CP does not conform to all properties of an adaptive model: it incorporates an explicit belief state, which is not typical for the adaptive approach to active vision. In the rest of this thesis, we shift our focus to models without a belief state. For example, in Chapter 4 we study how a memoryless model performs a task of gaze control for classification.

2.7.3 Probabilistic Models

The experiments also provide insight into the differences among the models of the probabilistic approach to active vision. Our results and analysis indicate that in most cases MI outperforms both L-ER and RK, while it does not take more actions than those models. Of course, performance is not the only factor of importance in choosing between these three models. In Section 2.3 we also discussed differences in training and execution time. For example, MI has no training time besides the estimation of the observation distributions, and has a long execution time. On the contrary, L-ER has a long training time, but a short execution time. RK has both a short training time and a short execution time, and it does not seem to perform much worse than L-ER.

2.7.4 Parameter Sensitivity

A drawback of L-ER and L-CP is that they involve many model parameters and training parameters (the number of hidden neurons, the number of policies, the mutation rate, the factor β, etc.)[6]. The long training time of the models prohibits an exhaustive search through the parameter space. In addition, when no successful policy can be learned, the reason for this often remains unclear. This raises the question to what extent the tuning of parameters forms an important practical obstacle. This could especially be so, if the results are highly sensitive to the parameter settings and if only a few settings lead to good results.

We want to show the influence of various model parameters and training parameters on the performance of the two learning models. As stated above, an exhaustive search through the parameter space is too time-consuming. Therefore, we take the parameters used for the experimental results as a starting point, and then change the parameters one by one. The experimental results then give an impression of the influence of the individual parameters on the classification performance. The model parameter varied is the number of hidden neurons. The training para-

[6]We note that the model MI does not have this drawback, since it has no model parameters that need to be trained. The model RK does not have this drawback either, because the training of its model parameters does not depend on any training parameters.

meters varied are: the number of policies λ, the number of selected policies μ and their offspring $\frac{\lambda}{\mu}$, the factor β, the mutation rate p_{mut}, and the number of evolutions.

Table 2.8: Influence of the different model parameters and training parameters on the performances of models L-ER and L-CP. Model PA is included in order to give an impression of the usual variation in experimental results. The mean performances (and standard errors of the mean) are based on 30 experimental runs. All parameter settings as used in the experiments with 25 objects in Section 2.5 are in bold italic.

Hidden neurons	0	5	***12***	50
L-ER	87.3 (1.3)	87.5 (1.4)	87.5 (1.6)	87.2 (1.6)
L-CP	88.5 (1.6)	89.2 (1.3)	88.7 (1.4)	89.5 (1.3)
PA	70.0 (1.9)	66.8 (2.1)	69.3 (2.2)	70.4 (2.1)
Number of policies (λ)	10	***20***	50	100
L-ER	87.7 (1.4)	87.5 (1.6)	87.1 (1.6)	87.7 (1.5)
L-CP	88.3 (1.5)	88.7 (1.4)	87.5 (1.7)	88.1 (1.3)
PA	70.0 (1.8)	69.3 (2.2)	69.2 (2.1)	67.7 (2.0)
Number of selected policies / offspring ($\mu / \frac{\lambda}{\mu}$)	1 / 20	***5 / 4***	10 / 2	
L-ER	88.1 (1.4)	87.5 (1.6)	88.1 (1.4)	
L-CP	88.3 (1.4)	88.7 (1.4)	88.4 (1.4)	
PA	69.3 (2.3)	69.3 (2.2)	69.3 (2.2)	
Factor β	1	2	***4***	6
L-ER	85.5 (1.7)	86.7 (1.6)	87.5 (1.6)	87.9 (1.4)
L-CP	78.7 (1.7)	89.1 (1.3)	88.7 (1.4)	87.2 (1.5)
PA	69.3 (2.3)	69.3 (2.3)	69.3 (2.2)	69.3 (2.3)
Mutation rate (p_{mut})	0.01	***0.04***	0.06	0.10
L-ER	87.3 (1.5)	87.5 (1.6)	87.9 (1.4)	87.6 (1.7)
L-CP	88.4 (1.4)	88.7 (1.4)	88.3 (1.4)	88.1 (1.5)
PA	69.3 (2.2)	69.3 (2.2)	69.3 (2.3)	69.3 (2.3)
Number of evolutions	1	2	***3***	4
L-ER	88.0 (1.2)	88.9 (1.4)	87.5 (1.6)	87.1 (1.5)
L-CP	88.5 (1.4)	86.9 (1.6)	88.7 (1.4)	88.3 (1.3)
PA	68.3 (1.8)	72.1 (2.2)	69.3 (2.2)	70.1 (2.9)

Table 2.8 shows the influence of various model parameters and training parameters on the performance of L-ER and L-CP for a subset size of 25 objects, taken from the tuning set. We also include the performance of the passive model, which does not depend on the parameters considered. It may give an impression of the usual variance in the experimental results. In addition, it indicates the level of difficulty of the subsets encountered in the experiments. The average performances and standard errors of the mean are based on 30 experimental runs. The parameter values that we used for the experimental results with 25 objects in Section 2.5 are shown in bold italic. The table shows that careful parameter tuning is not necessary for obtaining the results reported in this chapter. The performances of the models are not so sensitive to the parameter changes, except for parameter β. Both L-CP and L-ER profit from setting β to a value ≥ 2, especially L-CP. As a consequence of the limited sensitivity of the results to parameter changes, parameter tuning does not pose a problem for the turn-table task.

2.8 Chapter Conclusion

In this chapter, we focused on RQ 1: *How does the adaptive active vision approach relate to other approaches of active vision?* The overview of the literature showed that besides an adaptive approach, there is also a probabilistic approach to active vision. The probabilistic approach is different from the adaptive approach in that it uses a formal framework for action selection. Probabilistic models take actions that maximally reduce the uncertainty in their belief state. In contrast, adaptive active vision models learn to take actions on the basis of an external measure of success.

To answer RQ 1, we described six vision models in a common framework and compared them to each other empirically. The most important conclusion we may draw from the experimental results regards the adaptive model L-CP. One might have expected that its lack of a formal action strategy would have a negative influence on its performance. However, the experiments show that this is not the case. We may conclude that L-CP performs as well as or better than the probabilistic active vision models on the turn-table task.

This conclusion is important, since it encourages us to continue studying adaptive active vision models. In the next chapter, we introduce a framework for the adaptive active vision models studied in the remainder of the thesis.

Chapter

3

Gaze Control Framework

In Chapter 2 we compared different active vision models. The comparison showed that the adaptive model performed as well as or better than the probabilistic models. In the remainder of this thesis, the comparison with probabilistic active vision models continues to play a role in the background. Our main focus is on adaptive active vision models. We do not impose a belief state upon the models. Instead, we leave it to the self-organizing process of artificial evolution to develop a suitable internal state.

Furthermore, we will focus on active vision models employing *foveal vision*, i.e., which process only a part of the visual scene at a high resolution. These models are interesting for computer vision, because they have two advantages (Ballard, 1991):

1. they can simplify visual tasks by means of their actions,

2. they are computationally efficient.

The drawback of foveal vision models is that they cannot capture all relevant features from the visual field at the same time. Therefore, the active vision models have to control their *gaze* in an intelligent manner: they have to process those parts of the visual scene that are important to their task. We refer to these models as *gaze control models*.

In this chapter, we introduce a gaze control framework named ACT-FRAME. The gaze control models studied in the remainder of the thesis are all instantiations of ACT-FRAME, but are tailored to different tasks. In Section 3.1, we describe the three requirements that our gaze control framework imposes on the gaze control models. Then, in Section 3.2, existing gaze control models are discussed in the light of these requirements. Remark that the discussion of these models is different from our discussion of active vision models in the previous chapter, because not all existing gaze control models are active vision models. In Section 3.3 we describe ACT-FRAME. Finally, in Section 3.4, we discuss the implementation choices of the framework.

3.1 Three Requirements

The gaze control framework imposes three requirements on each gaze control model.

1. *The gaze control model takes local image samples as inputs, rather than processing the entire image.*
2. *The gaze control model has to be task dependent.*
3. *The gaze control model contains no assumptions on what gaze strategy to follow.*

These requirements are motivated by the following considerations. Requirement 1 is motivated by our goal of computational efficiency. A model that processes the entire image is not computationally efficient, because the relevant information is generally constrained to a small subset of image regions. In addition, such a model would be a passive vision model, which is not the focus of this thesis. Requirement 2 is also motivated by computational efficiency. A model that is not task dependent is not efficient, since it may be burdened by spurious elements and behaviours that are neither useful nor necessary for the task. Indeed, as mentioned in Chapter 1, human gaze behaviour is also task dependent. Requirement 3 is motivated by our desire to allow as many gaze strategies as possible. If we make assumptions on what gaze strategy the model should follow, we run the risk of excluding successful gaze strategies that are unknown to us.

3.2 Existing Gaze Control Models

There are many existing gaze control models, not all meeting our three requirements. Below we discuss the existing gaze control models in terms of which requirement they do *not* fulfil. We subdivide the models into four groups: the first three groups fail to fulfil the corresponding requirements explained above, and the fourth group does fulfil all requirements.

3.2.1 Failing Requirement 1

The first group consists of all gaze control models that do not fulfil the first requirement; they process the entire image. The group contains mostly models of attention that have as goal to predict human gaze locations in images (e.g., Itti, Koch, and Niebur, 1998; Privitera and Stark, 2000; Rajashekar *et al.*, 2003). As such, they do not aim at closely modelling the (active) gaze control process. Itti *et al.* (1998), Privitera and Stark (2000), and Brockmann and Geisel (2000) employed bottom-up measures of visual saliency to predict the gaze locations. Boccignone and Ferraro (2004), Rasolzadeh *et al.* (2006), Torralba *et al.* (2006), and Peters and Itti (2007) provided a way of combining the visual saliency with top-down, task-dependent signals. All the above-mentioned models constructed a list of gaze locations, ranked according to visual saliency or task importance. They did not actually perform any task and

can be regarded as a preprocessing stage of a final vision system. For example, Privitera and Stark (2000) used the gaze locations for efficient image compression.

3.2.2 Failing Requirement 2

The second group consists of all gaze control models that do not obey the second requirement; the models determine their gaze locations independently of the task. The models by Sela and Levine (1997), Rybak *et al.* (1998), Salah, Alpaydin, and Akarun (2001) entirely relied on visual saliency to determine the next gaze location. The model by Lee and Yu (1999) always gazed at locations that contained the most *bottom-up information*. Bottom-up information is correlated to the amount of variance in the observations, which does not depend on a particular task.

Models that only rely on visual saliency or bottom-up information are not efficient. They always look at the same locations if given the same image, and may spend processing time on locations that are not useful for their task.

3.2.3 Failing Requirement 3

The third group consists of all gaze control models that violate the third requirement; they make assumptions on what gaze strategy to follow. Henderson *et al.* (2001) completely determined the gaze strategy of their model, by manually predetermining the gaze locations for each image. As a consequence, gaze sequences were restricted to a limited number of fixed gaze locations. Rao *et al.* (1995), Terzopoulos, Rabie, and Grzesczuk (1996), Rao *et al.* (1997), Terzopoulos and Rabie (1997), and Smeraldi, Capdevielle, and Bigün (1999) created gaze control models for searching specific objects in images. Their gaze shifts were entirely determined by the image locations in the field of view that best resemble the object of interest. For example, Terzopoulos *et al.* (1996) and Terzopoulos and Rabie (1997) made a gaze control model for simulated fish. One of the behaviours of the fish was to follow orange fish. When they perceived the colour orange at the right border of their local visual inputs, they automatically directed their gaze to the right. The fish were not free to develop their own way of visually interacting with their environment. Young, Scott, and Bandera (1998) and Minut and Mahadevan (2001) also designed gaze control models for searching specific objects, but allowed the task to modulate the gaze strategy. Still, it was assumed that the gaze shifts should go to visually salient locations in the visual field. Existing gaze control models of the probabilistic approach to active vision (Bandera *et al.*, 1996; Klarquist and Bovik, 1998; Paletta, Fritz, and Seifert, 2005; Lacroix, Postma, and van den Herik, 2007) also followed clearly defined gaze strategies. Their action strategies were based on the assumption that gaze shifts serve to gather a maximal amount of information on a predefined world state. Sprague, Ballard, and Robinson (2007) studied a simulated person. It had to control its gaze in order to perform a task of walking on a path, while avoiding obstacles and gathering litter. The model learned to act on the basis of external rewards such as the amount of litter collected. It incorporated a belief state representing the probability of objects occurring at different distances and angles from the simulated person. In addition, the assumption was made that

the eye movements served to increase the certainty in the belief state, which is in violation with our third requirement.

3.2.4 Fulfilling All Requirements

The fourth group consists of adaptive gaze control models that fulfil our three requirements. We briefly describe the models in the group, along with their application domain. An early adaptive gaze control model was the one by Schmidhuber (1990). This model had to learn to find a black triangle in white images corrupted by various amounts of noise. Harvey *et al.* (1994) applied an evolutionary algorithm to a robot in the real world that had to approach a triangle and avoid a rectangle, both drawn on a wall of the arena in which it moved around. The works by Kato and Floreano (2001), Marocco and Floreano (2002), and Floreano *et al.* (2004) investigated a rather similar task: to differentiate black squares and triangles in images corrupted by various amounts of noise. Schlesinger and Parisi (2001), Schlesinger (2003), and Schlesinger and Casey (2003) used an adaptive active vision model in an abstract visual environment, in order to explain infant behaviour in response to occluded objects. In more recent work, adaptive gaze control models were applied to tasks of higher complexity, such as wall avoidance (Marocco and Floreano, 2002), simulated driving (Floreano *et al.*, 2004), and landmark navigation (Suzuki and Floreano, 2006a; Suzuki and Floreano, 2006b). The only task in which the gaze control model had to deal with natural images was in the wall-avoidance task. In this task a robot had to avoid walls in an empty arena, on the basis of its gaze control.

The gaze control models that we describe in the experimental chapters 4, 5, and 6, belong to the fourth group. The difference with other models in the fourth group mainly lies in the *application* of the models: we apply our models to tasks that are more challenging than the ones mentioned above. In Chapter 4 and 6 the tasks involve natural static images. In Chapter 5 we study a more difficult version of the driving task investigated by Floreano *et al.* (2004).

3.3 ACT-FRAME

The gaze control framework ACT-FRAME is inspired by the models in Kato and Floreano (2001), Marocco and Floreano (2002), Floreano *et al.* (2004). The models that instantiate ACT-FRAME have a closed loop of observations and actions. They take local image samples, and map these samples to gaze shifts across the image.

In Figure 3.1 we show the core principle behind ACT-FRAME. A model that instantiates the framework, has two interconnected modules. The first module is named *visual feature extraction* (see the bottom right dashed box in Figure 3.1). It takes a local sample at the current gaze location (x) and extracts visual features from this sample. Then it passes these features to the second module, named *controller* (see the top right dashed box in Figure 3.1). The controller processes the extracted features. It determines a gaze shift in the image on the basis of these features and (optionally) on the basis of its internal state. At the new gaze location (o), the process of feature extraction and gaze shifting is repeated. This iterative process continues until a stopping criterium is met.

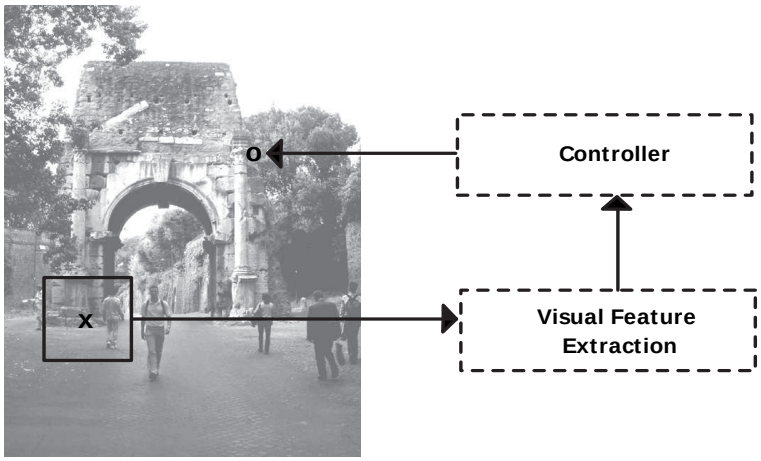


Figure 3.1: The gaze control framework, ACT-FRAME. A model based on the framework extracts visual features from a local image sample at the current fixation location (x). Then, its controller maps these feature values to a gaze shift in the image to the new fixation location (o). This iterative process continues until a stopping criterium is met.

3.4 Implementation Choices

By employing an adaptive approach, we attempt to let the active vision models as free as possible in the selection of a gaze control strategy. However, we cannot completely avoid the restriction of possible action strategies. Namely, we have to make implementation choices. For example, let us assume that we choose a feed-forward neural network as the model's controller. Such a network forms a memoryless mapping from the model's inputs to its outputs. Therefore, the choice for such a network restricts the model's gaze control strategies: it excludes strategies that make use of a memory[1].

Since implementation choices entail restrictions of the gaze control strategies, it is important to explain and motivate these choices. In this section, we explain our implementation choices regarding the visual feature extraction, the controller, and the optimisation algorithm used for adaptation.

3.4.1 Visual Feature Extraction

We make two main choices regarding the visual feature extraction. The first choice is to extract the visual features from a square window, referred to as the *gaze window*. Our motivation for the square shape of the window is that it facilitates the application of feature extraction methods from the field of computer vision[2]. We

[1]Remark that the models of the third group in Section 3.2 also incorporate implementation choices. However, they contain additional assumptions on which gaze control strategies to follow.

[2]Close modelling of human foveal vision with a highly detailed fovea and a coarse periphery is not our priority. Our focus is on the requirement that the gaze control model should process local image

choose the size of the gaze window in a task-dependent manner. For our choice we take into account that the window should be sufficiently large to perform the task, but sufficiently small to achieve computational efficiency and to necessitate intelligent gaze control.

The second choice concerns the manner in which features are extracted from the image. The feature extraction has to permit the gaze control model to handle tasks in natural images. Earlier models (such as Floreano *et al.*, 2004) used different feature extraction techniques such as taking the average gray value of an area or the gray-value of the centre pixel of an area. In Chapter 5 we study a task similar to the one in Floreano *et al.* (2004) and use the same feature extraction techniques. Since we want to address more difficult real-world problems involving natural images in Chapter 4 and 6, we employ an elaborate form of feature extraction in those chapters. In particular, we use the *integral features* introduced by Viola and Jones (2001).

3.4.2 Controller

In almost all of our experiments, we choose to use a neural network as the controller. The main four reasons for this choice are that: (1) a neural network can handle continuous inputs and continuous outputs, (2) a neural network resembles natural vision systems in the sense that it performs the task in a distributed manner, (3) a neural network can be extended so that it includes memory or other dynamic capabilities, and (4) it is a well-studied controller in robotics research.

For our experiments, we predefine a specific architecture for the neural network. There are methods that optimise both the network weights and the structure (e.g., Stanley and Miikkulainen, 2004; Mattiussi and Floreano, 2004). Although such methods have met with some success and may eventually be necessary, they are not yet guaranteed to give better solutions than methods that use a predefined structure (Floreano, Dürr, and Mattiussi, in press).

3.4.3 Optimisation Algorithm

For our adaptive approach to active vision, we can choose among different existing semi-supervised optimisation methods. Examples of such algorithms include random search, reinforcement learning (Sutton and Barto, 1998), simulated annealing (Kirkpatrick, Gelatt, and Vecchi, 1983), cross-entropy search (Rubinstein and Kroese, 2004), and evolutionary algorithms (Holland, 1992; Bäck, 1996). In our experiments we have used evolutionary algorithms.

We prefer evolutionary algorithms over random search, because the latter does not exploit any structure in the search space and is therefore highly inefficient.

Since there is no clear difference between evolutionary algorithms and reinforcement learning, we performed an experiment to compare the two techniques (de Croon, van Dartel, and Postma, 2005c). The outcome of the experiment is that evolutionary algorithms outperform reinforcement learning methods, especially when the visual inputs are ambiguous. This finding corresponds to other

samples.

findings in the literature (e.g., Gomez and Schmidhuber, 2005). In addition, evolutionary algorithms allow the optimisation of any part of the model that can be parametrised, while reinforcement learning focuses solely on the action strategy. This allows evolutionary algorithms to optimise the visual features and the action strategy simultaneously. A disadvantage of evolutionary algorithms with respect to reinforcement learning is that evolutionary algorithms themselves do not allow adaptation of a model while it is performing the task. However, we are more interested in how an adapted model handles a visual task, than in the adaptation process itself.

Evolutionary algorithms, simulated annealing, and cross-entropy search are three similar optimisation methods. Our choice for evolutionary algorithms is based on the fact that it is a common choice in the field of Evolutionary Robotics (Nolfi and Floreano, 2000).

In our experiments in Chapter 4, 5, and 6, we introduce gaze control models that instantiate the framework in different manners. We defer the explanation of the models' details to those chapters.

Chapter

4

Gaze Control and Sensory-motor Coordination

This chapter is based on parts of the following publications[1].

1. G.C.H.E. de Croon, E.O. Postma, and H.J. van den Herik (2005b). Sensory-motor coordination in gaze control. *Applications of Evolutionary Computing (EvoWorkshops 2005), Lausanne, Switzerland* (ed. F. Rothlauf), pp. 334 – 344, Springer-Verlag, (best paper award).
2. G.C.H.E. de Croon, E.O. Postma, and H.J. van den Herik (2005a). A situated model of active vision. *Belgian-Dutch AI Conference (BNAIC 2005), Brussels, Belgium* (eds. K. Verbeeck, K. Tuyls, A. Nowé, B. Manderick, and B. Kuijpers), pp. 74 - 80.
3. G.C.H.E. de Croon, E.O. Postma, and H.J. van den Herik (2006b). A situated model for sensory-motor coordination in gaze control. *Pattern Recognition Letters*, Vol. 27, No. 11, pp. 1181 - 1190.

In this chapter, we apply a gaze control model to a task of image classification. Since the adaptation process determines the model's action strategy, we do not know a priori what gaze control strategy the model will follow. In fact, as mentioned in Chapter 1, there is not yet a clear understanding of adapted active vision strategies. To improve our understanding of such strategies, we focus on RQ 2: *How does a memoryless adaptive gaze control model handle an image classification task?*

We investigate a *memoryless* gaze control model, because it facilitates analysis; each action of such a model only depends on the current observation. In the field of embodied cognitive science, the strategies of memoryless adaptive models are said to be based on *sensory-motor coordination* (Pfeifer and Scheier, 1999; Nolfi, 2002). We adopt the definition that sensory-motor coordination is the exploitation of an active model's closed loop of sensory inputs and motor actions to optimise the performance on a particular task[2]. To answer RQ 2, we compare a passive gaze control

[1]The author would like to thank the publishers and his co-authors for their permission to use parts of the publications in this chapter.

[2]We discuss a different definition in Section 4.6.

model (incapable of sensory-motor coordination) with an active gaze control model (capable of sensory-motor coordination). The passive gaze control model is named PAS-CLASS (passive classification) and the active gaze control model is named ACT-CLASS (active classification). The active gaze control model is an instantiation of ACT-FRAME.

We apply PAS-CLASS and ACT-CLASS to a task of gender recognition in static natural images. This task implies the classification of photos of human faces in two classes, male and female. The motivation for this task is three-fold: (1) the task is challenging, and no gaze control models have been applied to it so far; (2) the task involves a simulated model and static images, which saves optimisation time (Floreano *et al.*, 2004) and facilitates analysis (note that the use of a simulator preserves the possibility to study principles of sensory-motor coordination, cf. Pfeifer and Scheier, 1999); and (3) the task enables the comparison of two models that differ only in their ability to coordinate sensory inputs and motor actions.

The remainder of this chapter is organised as follows. First, we describe PAS-CLASS (Section 4.1) and ACT-CLASS (Section 4.2). Then, we explain our experimental setup (Section 4.3). Subsequently, we proceed in two phases. The first phase is to compare the performances of the two models (Section 4.4). For this comparison the performance difference between the models is more important than the absolute classification performance. Our only requirement is that the performances allow for a comparison. Hence, there should be no ceiling effect, in which both models have a high performance such as 98% and 99%. Neither should there be a floor effect, in which both models have a low performance such as 50% and 52%. If ACT-CLASS uses sensory-motor coordination, it should outperform PAS-CLASS. The second phase is to analyse how ACT-CLASS handles the classification task (Section 4.5). This analysis shows that ACT-CLASS succeeds in exploiting multiple observations for image classification, despite the memoryless nature of its controller. In Section 4.6 we discuss the relevance of the results. Finally, we present the chapter conclusion in Section 4.7.

4.1 PAS-CLASS

A passive model's main characteristic is that it always extracts visual inputs at the same locations in each image. It is an open-loop model: incoming inputs determine the classification, but not to which location the gaze is directed.

The passive model PAS-CLASS consists of three modules. The modules are illustrated by the dashed boxes in Figure 4.1: the visual feature extraction (I), the classifier (II), and the controller (III). We first give a short overview of the modules' functions, and then provide a detailed description of each module in Subsection 4.1.1 to 4.1.3.

Module I receives as sensory input the raw input (image gray values) from the gaze window with centre 'x', the current fixation location. In Figure 4.1 the raw input is shown on the left in box I; it contains a part of the face. From the gaze window, input features are extracted. These input features serve as input to module II, a neural network classifier. The input layer of the neural network is depicted by

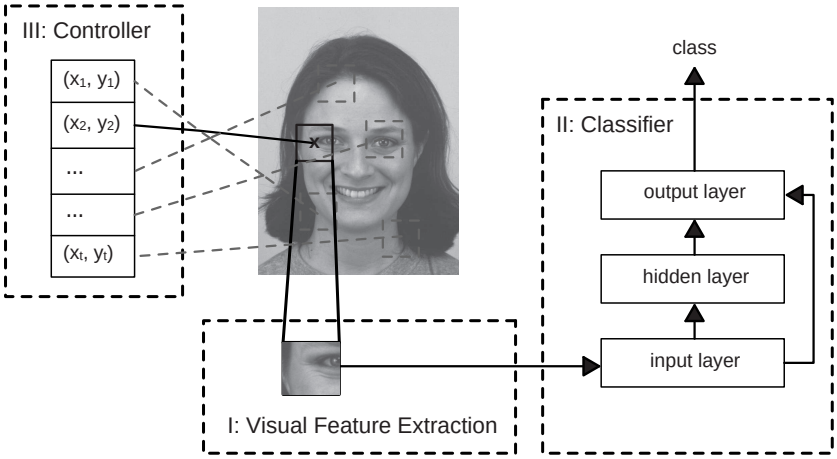


Figure 4.1: Overview of the passive model of gaze control.

the box 'input layer'. Subsequently, the neural network calculates the activations of the hidden neurons in the 'hidden layer' and of the output neuron in the 'output layer'. There is one output neuron that indicates the class of the image. Module III (left in Figure 4.1) is the controller that determines the next fixation location, where the process is repeated. Below we describe the three modules of PAS-CLASS in more detail.

4.1.1 Visual Feature Extraction

The first module performs the visual input feature extraction. For our research, we adopt the type of features as introduced in Viola and Jones (2001).

The module extracts ten input features from the gaze window. Each input feature represents the difference in mean light intensity between two areas in the window. The areas are determined by the feature's type and location. Figure 4.2 shows eight different feature types (top row) and nine differently sized locations in the gaze window from which the input features can be extracted (middle row, left). The sizes vary from the entire gaze window to a quarter of the gaze window. In total, there are $8 \times 9 = 72$ different input features. In Figure 4.2, two example input features are given (middle row, right). Example feature 'L' is a combination of the first type and the second location, example feature 'R' of the third type and the sixth location. The bottom row of the figure illustrates how an input feature is calculated, namely by subtracting the mean light intensity in the image covered by the gray surface (area 'A') from the mean light intensity in the image covered by the white surface (area 'B'). The result is a real number in the interval $[-1, 1]$. In the case of example feature L, only the left half of the gaze window is involved in the calculation.

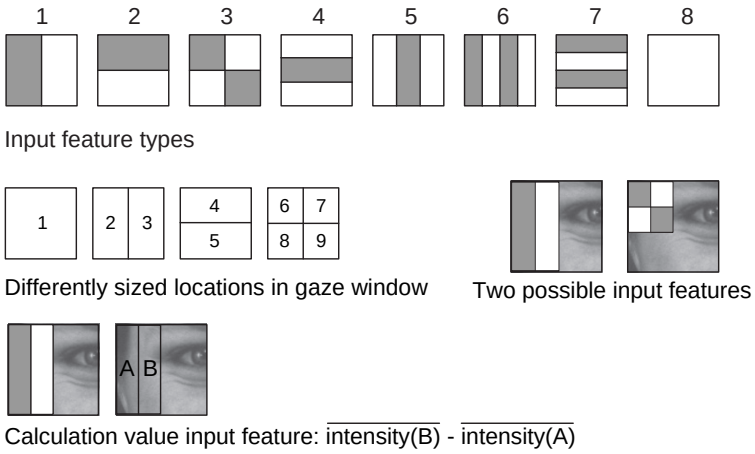


Figure 4.2: An input feature consists of a type and a location. **Top:** Different input feature types. **Middle (left):** The different sizes and positions at which the features can be extracted. **Middle (right):** Two example combinations of input feature types and sizes / locations. **Bottom:** The value of an input feature is calculated by subtracting the mean light intensity in the image under the gray area from that covered by the white area.

4.1.2 Classifier

The second module is a neural network classifier that takes the extracted input features as inputs. It is a fully connected feedforward neural network with $h = 3$ hidden neurons and one output neuron. The hidden and output neurons all have sigmoid activation functions: $a(z) = \tanh(z) = 1 - \frac{2}{1+e^{2z}}$, $a(z) \in \langle -1, 1 \rangle$. The activation of the output neuron (out_1) determines the classification. A classification is correct if $\text{sign}(\text{out}_1) = c$, with $c = 1$ for male and $c = -1$ for female. The sign-function takes on the following values: $\text{sign}(z) = 1$ if $z \geq 0$ and $\text{sign}(z) = -1$ if $z < 0$. Since preliminary experiments showed that evolved weights were often close to 0, the neural network weights are constrained to the interval $[-r, r] = [-1, 1]$.

4.1.3 Controller

The third module is a controller that consists of a list of fixation locations visited in every image. PAS-CLASS first shifts its gaze to the first location in the list, (x_1, y_1), and then classifies the image. Subsequently, it fixates the next location, (x_2, y_2), and again classifies the image. This process continues, until PAS-CLASS has fixated all locations from (x_1, y_1) to (x_t, y_t) in sequence, assigning a class to the image at every fixation. The performance is based on these classifications. The t fixation locations are selected by an evolutionary algorithm (see Subsection 4.3.2). Selecting the fixation locations also implies selecting the order in which they are fixated.

4.1.4 Adaptable Model Parameters

PAS-CLASS has four types of model parameters that are to be adapted to the task by an evolutionary algorithm:

1. the types and places of the input features,
2. the size of the gaze window from which features are extracted,
3. the neural network weights,
4. the coordinates of all fixation locations.

4.2 ACT-CLASS

The active gaze control model is an instantiation of ACT-FRAME. Its main characteristic is that it determines its gaze shifts on the basis of its observations. It iteratively takes a local sample from the image at the current fixation location, extracts features from this sample, and then determines the next fixation location. Hence, ACT-CLASS has a closed loop of sensory inputs and actions, in contrast to PAS-CLASS. This closed loop permits ACT-CLASS to influence subsequent observations on the basis of the current one.

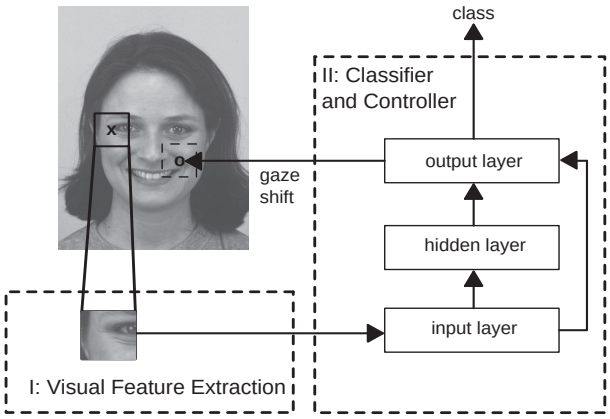


Figure 4.3: Overview of the active model of gaze control.

4.2.1 The Modules

Figure 4.3 shows an overview of the active gaze control model. It consists of two modules. Module I is equal to that of the passive vision model: it extracts ten feature values and passes these to the second module. Module II is a combination of

a classifier and a controller. It has the same number of input neurons and hidden neurons as that of PAS-CLASS, but it maps the input values to both a classification *and* a gaze shift. Hence, it has three output neurons. The first output neuron indicates the classification as explained above. The second and the third output neurons determine a gaze shift (Δx, Δy) as follows.

$$\Delta x = \lfloor d_{\text{max}} \times \text{out}_2 \rfloor \tag{4.1}$$

$$\Delta y = \lfloor d_{\text{max}} \times \text{out}_3 \rfloor \tag{4.2}$$

The variables out_2 and out_3 represent the activations of the second and third output neurons, respectively. Moreover, d_{max} is the maximum number of pixels that the gaze can shift in the x- or y-direction. As a result, Δx and Δy are expressed in pixels. We set $d_{\text{max}} = 500$, so that the model can reach almost all locations in the image in one time step. If a shift brings the gaze window over the border of the image, the fixation location is repositioned to the nearest possible fixation location. In Figure 4.3 the current fixation location is represented by an 'x', and the new fixation location as determined by the neural network by an 'o'.

4.2.2 Adaptable Model Parameters

ACT-CLASS has three types of model parameters that are to be adapted by an evolutionary algorithm:

1. the types and places of the input features,
2. the size of the gaze window from which features are extracted,
3. the neural network weights.

4.3 Experimental Setup

In this section, we describe the gender recognition task used for the comparison of PAS-CLASS and ACT-CLASS. In addition, we discuss the evolutionary algorithm that optimises the adaptable parts of both models.

4.3.1 Gender Recognition Task

We chose the image classification task of gender recognition, since it is a challenging and well-studied task (Bruce and Young, 2000). The task consists of classifying photos as containing a female or male face. There are many differences between female and male faces that can be exploited by gender recognition models (Moghaddam and Yang, 2002; Calder *et al.*, 2001). State-of-the-art models use global features, extracted in a passive manner. So far, none of the current models is based on gaze control and local image samples.

The task for the models is to determine whether an image contains a photo of a male or female face. They base their classification on the input features extracted

from the gray-scale images at the fixation locations. PAS-CLASS shifts its gaze to the sequence of t fixation locations as determined by its third module. In contrast, ACT-CLASS starts at the centre of the image and determines its subsequent $t-1$ fixation locations with output values of the neural network. At every fixation location, the models extract features from the image and use these features to assign a class to the image.

PAS-CLASS and ACT-CLASS are optimised on a training set of images, and tested on a separate test set of images. The data set for the experiment was constructed by J.E. Litton of the Karolinska Institutet in Sweden. It contains 276 images with angry-looking and happy-looking human subjects. These images are converted to gray-scale images and resized to 600×800 pixels. One half of the image set serves as a training set, the remaining half of the image set is used as the test set to determine the performance of the optimised gaze control models. Both training set and test set consist of 50% males and 50% females.

4.3.2 Evolutionary Algorithm

We optimise the adaptable parameters of each model with a λ, μ-evolutionary algorithm (Bäck, 1996). The genome represents the four types of adaptable model parameters as follows.

1. For both PAS-CLASS and ACT-CLASS the genome represents the types and places of the input features, each by one double value. When decoding the genome, we first map the double value to a binary string of length 7. The first two bits encode for the feature's scale (large square, wide rectangle, high rectangle, small square). The following two bits encode for the place, if the feature is sufficiently small to fit in the window (top left, top right, bottom left, bottom right). A high rectangle with as place 'bottom right' will occupy the right half of the window. The last three bits encode for the eight possible feature types.

2. For both PAS-CLASS and ACT-CLASS the genome represents the size of the gaze window by one double value, $s \in [0, 1]$. The side of the window has length $50 + \lfloor 100s \rfloor$. Preliminary experiments showed that this range of scales is sufficiently large to perform the gender recognition task, and sufficiently small to necessitate intelligent gaze control.

3. For both PAS-CLASS and ACT-CLASS the genome represents the neural network weights, each by one double value in the range $[-r, r] = [-1, 1]$.

4. For PAS-CLASS the genome represents a list of fixation locations that is used for every image. Each fixation location is represented by two double values $v_{i1}, v_{i2} \in [0, 1]$. They encode a fixation location as follows: $(x_i, y_i) = (\lfloor (549 - \lfloor 100s \rfloor) \times v_{i1} \rfloor + 1, \lfloor (749 - \lfloor 100s \rfloor) \times v_{i2} \rfloor + 1)$. The location (x_i, y_i) represents the top left coordinate of the gaze window.

In our experiments, we execute 15 independent *evolutionary runs* to obtain a reliable estimate of the average performance. Each evolutionary run starts by creating

an initial population of $\lambda = 30$ randomly initialised models. Each model operates on every image in the training set, and is evaluated on the basis of its average classification performance over all time steps and images. This is expressed by the following fitness function:

$$f = \frac{\sum_{j=1}^{n} \sum_{i=1}^{t} \delta(\text{sign}(\text{out}_1(i)), c_j)}{n \times t}, \tag{4.3}$$

in which $\text{out}_1(i)$ is the activation of the first output neuron at time step i, c_j is the class of image j, and δ is the Kronecker delta ($\delta(a, b) = 1$ if $a = b$ and $\delta(a, b) = 0$ if $a \neq b$). Furthermore, n is the number of images in the training set, and $t = 5$ is the total number of time steps (fixations) per image. We note that the product $n \times t$ is a constant that normalises the performance measure. The $\mu = \frac{\lambda}{2} = 15$ models with the highest performance are selected to form the population of the next generation. Their adaptable parameter sets are mutated with probability $p_{\text{feat}} = 0.02$ for the input feature parameters and $p_{\text{gene}} = 0.10$ for the other parameters, e.g., representing coordinates or network weights. If mutation occurs, a feature parameter is perturbed by adding a random number drawn from the interval $[-v_{\text{feat}}, v_{\text{feat}}] = [-0.5, 0.5]$. For other types of parameters, this interval is $[-v_{\text{gene}}, v_{\text{gene}}] = [-0.1, 0.1]$. The evolution stops after $g = 300$ generations.

4.4 Performance

As stated before, we executed 15 evolutionary runs separately for PAS-CLASS and for ACT-CLASS. For each evolutionary run, we selected the instance that performs best on the training set, and determined its performance on the test set. The table in the left part of Figure 4.4 shows the mean performances and the standard deviation on the test set of the best instances of PAS-CLASS and ACT-CLASS of the evolutionary runs. Performance is expressed as the proportion of correct classifications.

ACT-CLASS outperforms PAS-CLASS. The table shows that for both tasks, the mean performance of the best instances of ACT-CLASS is higher than that of the best instances of PAS-CLASS. The distributions of the performances are highly skewed, as shown in the right part of Figure 4.4 (gray bars for PAS-CLASS, black bars for ACT-CLASS). Therefore, we applied a randomisation test (Cohen, 1995) to verify the statistical significance of the results. It revealed that the difference between the mean performances of the two types of models is significant ($p < 0.05$).

The performance difference between PAS-CLASS and ACT-CLASS is not so sensitive to the specific parameter settings. As an illustration, we investigated the sensitivity of the performance difference to the task parameter t, the total number of time steps. Figure 4.5 shows the mean performance of instances of PAS-CLASS (square markers, gray) and ACT-CLASS (circular markers, black) for $t \in \{5, 10, 15, 20\}$. It also shows the standard errors for the mean performances. All results are based on 15 evolutionary runs. ACT-CLASS significantly outperforms PAS-CLASS for all settings of t.

Gender Task	$\bar{f}\ (\pm\sigma)$
PAS-CLASS	$0.60(\pm 0.06)$
ACT-CLASS	$0.75(\pm 0.06)$

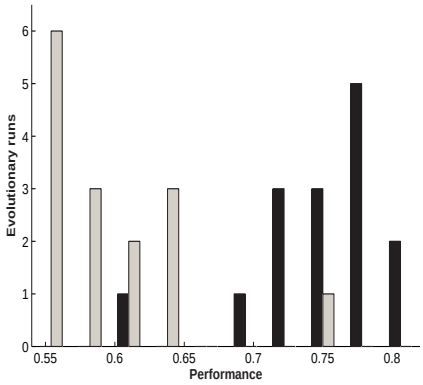


Figure 4.4: Performances of the best instances of their evolutionary run of PAS-CLASS and ACT-CLASS. **Left:** Mean performance ($\bar{f}$) and standard deviation (σ) of the performance on the test set of the best instances of the 15 evolutionary runs. **Right:** Histograms of the best performance of each evolutionary run. Gray bars are for instances of PAS-CLASS, black bars for instances of ACT-CLASS.

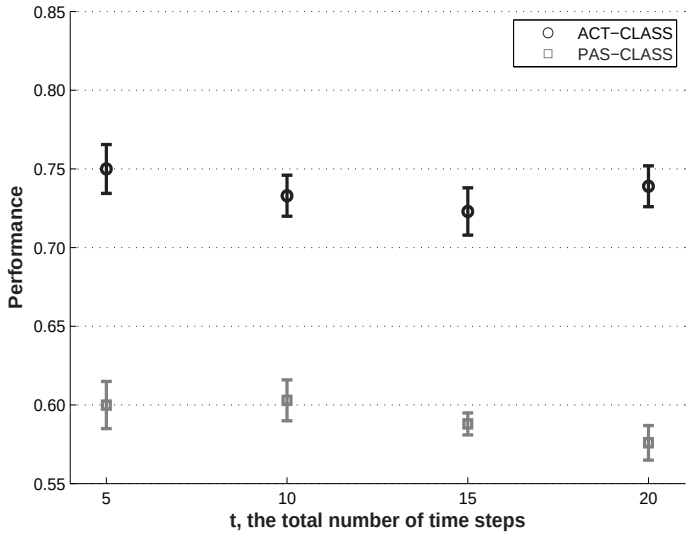


Figure 4.5: Influence of the task parameter t on the mean performance of passive models (square markers) and active models (circular markers).

4.5 Analysis

In this section, we investigate the precise reasons why ACT-CLASS outperforms PAS-CLASS. The reason for the performance difference has to be related to sensory-motor coordination, since this is the main difference between the models. Therefore, we analyse *how* ACT-CLASS uses sensory-motor coordination to improve its

performance on the gender recognition task. For this analysis, we select the best instance of ACT-CLASS out of all evolutionary runs of the gender recognition task (with $t = 5$). The analysis will show that sensory-motor coordination allows ACT-CLASS to exploit the class structure and the specific structure of the images to increase the performance over time.

4.5.1 Probabilistic View on an Adaptive Model

We would like to analyse sensory-motor coordination in a formal manner. In previous studies on sensory-motor coordination in classification tasks (Scheier *et al.*, 1998; Nolfi and Marocco, 2002), the Geometric Separability Index (GSI) was used as a formal measure. The GSI is a measure of how well two classes are linearly separable in the input space. Hence, the drawback of this measure is that it is only suited for linear classifiers and for binary classification tasks. So, we did not use the GSI.

In our analysis of the mechanism of sensory-motor coordination, we adopt a probabilistic framework. This is a natural choice, since the classification task involves uncertainty. Furthermore, a probabilistic framework allows us to use Shannon's (1948) entropy as a formal measure for the success of sensory-motor coordination. This measure does not have the mentioned drawback of the GSI. In addition, it facilitates the interpretation of sensory-motor coordination with respect to the probabilistic approach to active vision. We emphasise that we use a probabilistic framework for *analysis* purposes, while active vision models of the probabilistic approach use it to determine their actions (see Chapter 2).

ACT-CLASS' module II is a neural network that combines the functions of classifier and controller. However, for our probabilistic analysis it is better to view these functions separately. Hence, we will refer to the function represented by the neural network that maps an input feature vector to a class as the model's *classifier* and the function that maps an input feature vector to a gaze shift as the model's *controller*. We remind that PAS-CLASS has a separate classifier (a neural network) and controller (a list of fixation locations).

Below, we first explain the role of the classifier of both PAS-CLASS and ACT-CLASS, and then the role of their controller. Subsequently, we indicate the difference between PAS-CLASS and ACT-CLASS that may explain their performance difference.

Classifier

For both PAS-CLASS and ACT-CLASS the classifier is evolved on the basis of its classification of the input feature vectors at all time steps i. The evolutionary algorithm has to find a mapping from an observation o_{ij} to the class of image j, $c_j \in \{-1, 1\}$. Bayes' rule allows finding a suitable mapping from o_{ij} to c_j. It states that for all different classes c: $p(c \mid o_{ij}) = p(o_{ij} \mid c)\, p(c)/p(o_{ij})$, where $p(c \mid o_{ij})$ is the probability of class c, given a single observation o_{ij}. This probability is referred to as the *posterior probability*. Classification can be done by finding the class with maximal

posterior probability[3]. In our experiments the evolutionary algorithm optimises the classifier in order to maximise the classification performance on the training set. As a consequence, we expect that the classes selected by the classifier correspond to the classes with **M**aximal **A P**osteriori (MAP) probability. In other words, we expect the classifier to approximate a MAP classifier.

Controller

To understand the role of the controller of both PAS-CLASS and ACT-CLASS, the notion of entropy is of importance. The performance of a MAP classifier on our gender recognition task depends on the entropy of the posterior probabilities for the two classes. A maximal entropy indicates that given the observation, all classes are equally probable. A minimal entropy indicates that given the observation, only one class is possible. A lower entropy of the posterior probabilities improves the performance of the classifier. Therefore, the controller can improve the performance by taking actions that decrease the entropy of the posterior probabilities of the classes.

As mentioned, the probabilistic active vision models discussed in Chapter 2 make it an explicit goal to minimise the expected entropy of the belief state by performing multiple observations. The belief state in such models is $p(c \mid o_{1j}, o_{2j}, \ldots, o_{tj}, a_{1j}, a_{2j}, \ldots, a_{tj})$. The active model studied in this chapter does not incorporate a belief state. In fact, the feedforward neural network classifier employed in our experiments has no obvious means of optimising $p(c \mid o_{1j}, o_{2j}, \ldots, o_{tj}, a_{1j}, a_{2j}, \ldots, a_{tj})$, since it only receives the current observation. Therefore, we expect the controller to minimise the entropy of the posterior probabilities $p(c \mid o_{ij})$, i.e., of a single observation.

Difference PAS-CLASS and ACT-CLASS

What is the difference between PAS-CLASS and ACT-CLASS that can explain their performance difference? Both PAS-CLASS and ACT-CLASS can minimise the entropy of the posterior probabilities of a single observation by means of choosing fixation locations. Since both models have the same classifier, the cause of the better performance of ACT-CLASS must be in the selection of better fixation locations.

In contrast to PAS-CLASS, ACT-CLASS can influence subsequent observations on the basis of the current observation. Below, we show that this influence allows ACT-CLASS to exploit multiple observations in spite of the memoryless nature of its neural network. The exploitation of multiple observations leads to a decrease of the entropy of the posterior probabilities over time.

In our analysis we mainly focus on ACT-CLASS. The analysis consists of four steps. First, we demonstrate that ACT-CLASS shifts its gaze to better locations for each class, based on multiple observations (Subsection 4.5.2). Second, we show that ACT-CLASS does the same for specific images (Subsection 4.5.3). Third, we demonstrate that the gaze behaviour of ACT-CLASS leads to an increase of performance over time (Subsection 4.5.4). Fourth, we measure the entropy of the posterior

[3]The class with maximal posterior probability is usually found by maximising the likelihood: $c_j = \text{argmax}_{c \in C} p(o_{ij} \mid c)$, under the assumption that $p(c)$ is equal for every class.

probabilities of single observations over time (Subsection 4.5.5). Our measurements show that ACT-CLASS reduces the entropy of the posterior probabilities over time, while PAS-CLASS does not.

4.5.2 Gaze Behaviour per Class

In this subsection we present some empirical evidence for our suggestion that ACT-CLASS' controller gathers observations that minimise the entropy of the posterior probabilities (i.e., observations that are easier to classify).

We provided the best instance of ACT-CLASS with observations typical for male or female images and then analysed the resulting gaze path to assess class-dependent behaviour. The analysis shows that the active model (1) fixates locations at which its classifier performs well, i.e., where the entropy of the posterior probabilities is low, and (2) exploits multiple observations. ACT-CLASS obtains different observations for male and female images by fixating different locations for each class. Below, we demonstrate the above two properties of ACT-CLASS' gaze strategy by first determining the performance of its classifier at different locations in the image, and then studying ACT-CLASS' gaze behaviour.

Performance at Different Locations

To obtain an impression of the fixation locations at which the classifier performs well for male or for female images, we measured ACT-CLASS' classification performance on the training set at all positions of a 100×100 grid superimposed on the image. At every position we determined the classification performance for both classes. The *local average performance* for a class c is the proportion of the images of that class that is correctly classified by the model at coordinate (x, y). The local average performance $l_c(x, y)$ is defined as follows.

$$l_{\mathrm{c}}(x, y) = \frac{\sum_{j=1}^{n} \delta(c_j, c)\delta(\mathrm{sign}(\mathrm{out}_1(x, y)), c_j)}{\sum_{j=1}^{n} \delta(c_j, c)} \tag{4.4}$$

This local average performance is closely related to the entropy of the posterior probabilities: high performance corresponds to low entropy and vice versa. The left part of Figure 4.6 shows a picture of the local average performances for male images ($l_1(x, y)$) represented as intensities for all locations. The highest intensity represents perfect classification. The left part of Figure 4.7 shows the local average performances for female images ($l_{-1}(x, y)$). The figures show that dark areas in Figure 4.6 tend to have high intensity in Figure 4.7 and vice versa. Hence, there is an obvious trade-off between good classification of males and good classification of females[4]. The presence of a trade-off implies that classification of males and females should ideally take place at different locations.

[4]Note that the images are not inverted copies: in locations where male and female inputs are quite different, good classification for both classes can be achieved.

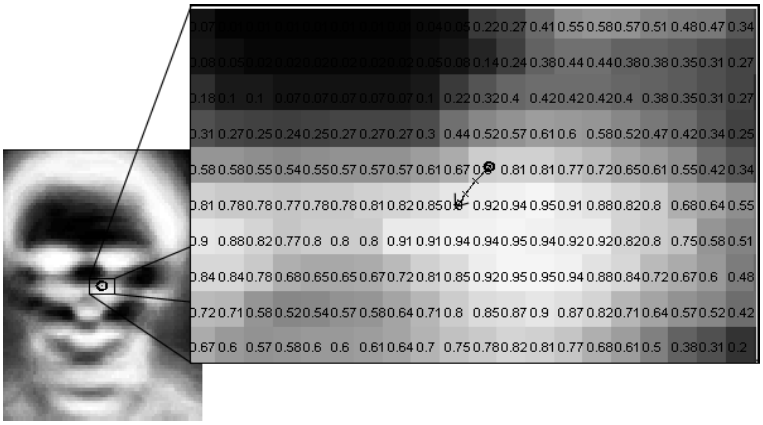


Figure 4.6: Local average performances on male images in the training set. The arrow superimposed on the large inset represents the gaze path of the active model, when it receives inputs typical for male images.

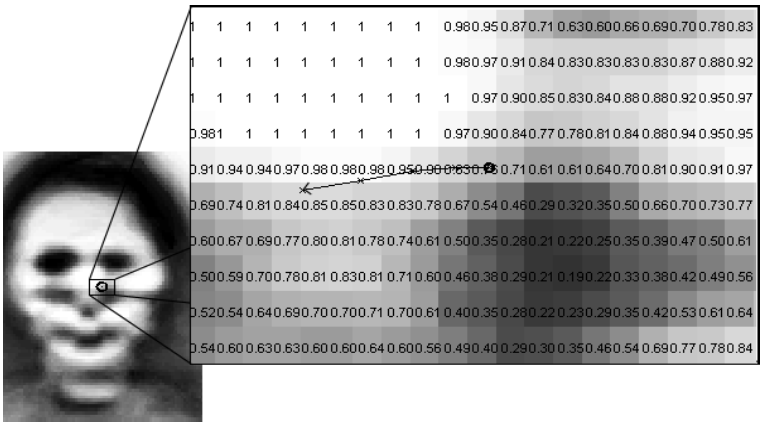


Figure 4.7: Local average performances on female images in the training set. The arrow superimposed on the large inset represents the gaze path of the active model, when it receives inputs typical for female images.

Gaze Behaviour

Below, we analyse the gaze path that originates when we provide ACT-CLASS with either typically male observations at all fixation locations or typically female observations at all fixation locations. To determine a 'typically male observation' at a given location, we extract the input feature vector at that location for every male image in the training set and calculate the average input feature vector. To determine a 'typically female observation' we follow the same procedure, but for the female images in the training set.

The right part of Figure 4.6 zooms in on the picture and shows the gaze path

that results when ACT-CLASS receives typical male inputs at all fixation locations. The first fixation location is indicated by an 'o'-sign, the last fixation location by an arrow. Intermediate fixations are represented with the 'x'-sign. The black lines in Figure 4.6 connect the fixation locations. The active model moves from a region with performance 0.80 to a region with performance 0.90. The right part of Figure 4.7 shows the same information for images containing females, revealing a movement from a region with a performance of 0.76 through a region with a performance of 0.98.

Both figures show that ACT-CLASS *fixates locations at which its classifier performs well on the class associated with the observations (1)*. They also show that the model takes misclassifications into account: it avoids areas in which the performance for the alternative class are too low. For example, if we look at the right part of Figure 4.6, we see that ACT-CLASS fixates locations to the bottom left of the starting fixation, while the local average performance is even higher to the bottom right. The reason for this behaviour is that in that area the performance for female images is rather low (Figure 4.7).

ACT-CLASS fixates different locations for different observation histories (typically male or typically female). Therefore, the fixation location is a form of *external memory*[5]: it encodes for the history of past observations. The model reaches better classification locations by making multiple gaze shifts based on multiple observations. This implies that ACT-CLASS *exploits multiple observations to optimise its performance (2)*.

Passive models cannot exploit multiple observations to fixate better classification areas, since the fixation locations are determined in advance for all images. As a consequence, PAS-CLASS cannot fixate different locations for the two classes.

4.5.3 Gaze Behaviour per Specific Image

We showed that ACT-CLASS exploits the differences between the two classes to select its fixation locations. In this subsection we show that it also exploits the differences between specific images to select its fixation locations. For specific images, ACT-CLASS often deviates from the general gaze path, venturing even into regions of the image that have low local average performances. For specific images that differ considerably from the average, these areas might be well-suited for classification. Below, we first give an intuitive example, and then provide some empirical evidence that ACT-CLASS exploits the structure of specific images.

Intuitive Example

Our example concerns the gaze shifts of ACT-CLASS to locate the eyebrows of a person in a specific image. The model partly bases its classification on the eyebrows. However, in some images the eyebrows are lifted, so that they occur higher in the image than usual. In this case, ACT-CLASS fixates a location right and above of the

[5]With 'external' we mean external from the viewpoint of the controller and classifier, not necessarily external from the viewpoint of ACT-CLASS. We remark that it is debatable whether the fixation location is external to ACT-CLASS or not.

starting fixation. On average, this location is not good for male classification (its local average performance is 0.57, see Figure 4.6), since in our training set eyebrows are usually below this location. Yet, for specific images with lifted eyebrows it *is* a good area. Since ACT-CLASS only fixates this area if the eyebrows are lifted, the actual performance for such images is higher than the performance we would predict on the basis of the local average performance.

Empirical Evidence

Moreover, we provide empirical evidence that ACT-CLASS exploits the structure of specific images by comparing its performance with a predicted performance that is based on the local average performances. The predicted performance of ACT-CLASS at time step i in image j is defined as $l_{c_j}(x_{ij}, y_{ij})$, where (x_{ij}, y_{ij}) is the fixation location of ACT-CLASS. As a consequence, the predicted performance is based on the assumption that ACT-CLASS only uses class-specific information to select good classification areas. If the actual performance of ACT-CLASS is higher than the predicted performance, it exploits more than just this class-specific information.

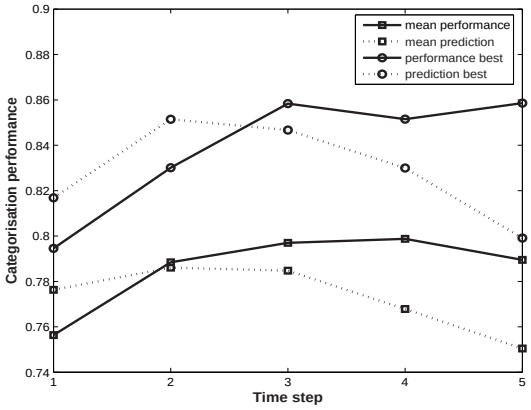


Figure 4.8: Actual performance (solid lines) and the predicted performance (dotted lines) over time, for the best instance of ACT-CLASS in particular (circular markers), and averaged over all instances of ACT-CLASS (square markers).

Figure 4.8 shows the actual performance and the predicted performance. The actual performance is indicated with solid lines and the predicted performance with dotted lines. We plot this information for the best instance of ACT-CLASS in particular (circular markers) and averaged over all 15 instances of ACT-CLASS that were best of their evolutionary run (square markers). For the last three time steps, the actual performances are consistently higher than the predicted performances. Apparently, the actual performance at a given location is better than the average performance at that location. Hence, ACT-CLASS exploits the structure of specific images.

4.5.4 Performance over Time

The final result of the gaze behaviours of all instances of ACT-CLASS is the optimisation of the (actual) performance over time. Figure 4.8 shows that the actual performance increases after $i = 1$. The fact that the performance generally increases over time suggests that sensory-motor coordination establishes dependencies between multiple actions and observations that serve to optimise the classification performance.

4.5.5 Entropy over Time

In Subsection 4.5.1, we stated that ACT-CLASS minimises the entropy of the posterior distribution over time, while PAS-CLASS does not. Below, we corroborate our statement by estimating the entropy of the posterior distribution at each time step. We first make an estimate for the best instance of ACT-CLASS alone. Then, we also estimate the entropy over time for the other instances of ACT-CLASS that were the best of their evolutionary run, and for all best instances of PAS-CLASS.

Entropy of Best Instance of ACT-CLASS

We estimate the entropy of the posterior distribution at each time step for the best instance of ACT-CLASS. By treating ACT-CLASS' observations as elements of a set O, we calculate the conditional entropy (cf. Shannon, 1948) of the posterior distribution:

$$H(C \mid O, i) = \sum_{o \in O} p(o_i)\, H(C \mid o_i), \tag{4.5}$$

$$H(C \mid o_i) = \sum_{c \in C} p(c \mid o_i) \log_2 \left(\frac{1}{p(c \mid o_i)} \right), \tag{4.6}$$

in which $p(o_i)$ is the probability of observation o at time step i and $H(C \mid o_i)$ is the Shannon entropy of the posterior probability distribution for observation o at time step i. Furthermore, C is the set of all mutually exclusive classes c.

In order to obtain reliable estimates of $p(o_i)$ and $p(c \mid o_i)$, we reduce the large number of input feature vectors by mapping them onto a limited set of prototypes. This mapping is performed with k-means clustering (see, e.g., Jain, Murthy, and Flynn, 1999). The value of k indicates the number of clusters and is therefore proportional to the number of input feature vectors that are mapped to a cluster. To obtain reliable estimates, we select small values of k. We have performed k-means clustering with a Euclidian distance measure for various choices of k.

Figure 4.9 shows the performance and the entropy over time for $k = 10$ and $k = 15$. Please note that the y-axis represents both (1) the proportion of correctly classified images for the performance (solid line), and (2) the entropy expressed in bits (dashed line for $k = 10$ and dotted line for $k = 15$). For each setting of

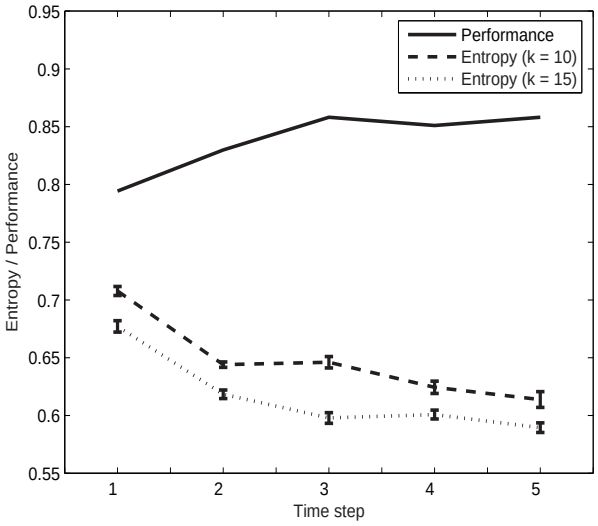


Figure 4.9: Entropy over time of the best instance of ACT-CLASS averaged over 30 runs of the k–means algorithm for $k = 10$ (dashed line) and $k = 15$ (dotted line). The performance over time is also shown (solid line).

k we obtain the entropy over time, by taking the average over 30 runs of the k-means algorithm. The figure also includes the standard errors associated with these average values.

The performance and the entropy of the posterior distribution should be inversely related, as was explained in Subsection 4.5.1. Indeed, Figure 4.9 illustrates that the entropy of the posterior distribution decreases over time, while the performance increases over time. The entropy over time for $k = 15$ best matches the data of the actual performance, since for $k = 15$ the entropy increases slightly from $i = 3$ to $i = 4$. This coincides with a slight drop in performance from $i = 3$ to $i = 4$. We included $k = 10$ to show that for other values of k the entropy may not closely match the performance graph, but that the general observation of a decreasing entropy always holds (other values of k give similar results).

Since our entropy measurements are based on the clusterings that we performed on the inputs, one can argue that it is not very exact. Therefore, we have also performed experiments in which the active gaze control models receive discrete observations from an observation set O (de Croon, Postma, and van den Herik, 2005a)[6]. The discrete observations allowed an exact calculation of the entropy of the posterior distribution and of the performance of the MAP classifier. The experimental results confirmed our expectation that the evolved active models' classifiers perform as MAP classifiers. In addition, the evolved active models achieved the same decrease in entropy over time as the instance of ACT-CLASS discussed above.

[6]See Subsection 4.6.1 for a short discussion of this model in the light of the Markov assumption.

Entropy of PAS-CLASS and ACT-CLASS

The entropy does not only decrease for the best instance of ACT-CLASS of all evolutionary runs. Figure 4.10 shows the entropy over time ($k = 10$) for both PAS-CLASS and ACT-CLASS, averaged over all 15 best instances of the evolutionary runs. It illustrates that, on average, the entropy decreases over time for the instances of ACT-CLASS until $i = 4$. The average entropy seems to decrease (non-monotonously) for the instances of PAS-CLASS as well, but the magnitudes of the error bars indicate that the entropies vary significantly. In fact, the entropies of the instances of PAS-CLASS do not decrease or increase reliably over time.

The fundamental difference between PAS-CLASS and ACT-CLASS is that only the latter can decrease the entropy of the posterior distribution over time. Figure 4.10 shows that there is an additional difference that accounts for PAS-CLASS being outperformed by ACT-CLASS: there is already a considerable entropy difference at the first time step. The evolutionary algorithm seems to have difficulties reaching the optimal instantiation of PAS-CLASS. Figure 4.4 demonstrates that such difficulties exist, since one of the evolutionary runs has succeeded in obtaining a performance significantly higher than the other evolutionary runs. The best instantiation of PAS-CLASS has a performance of 0.76, while the other instantiations have performances in the range of 0.56 to 0.64. Nonetheless, the best instantiation of PAS-CLASS has a lower performance than the best instantiations of ACT-CLASS.

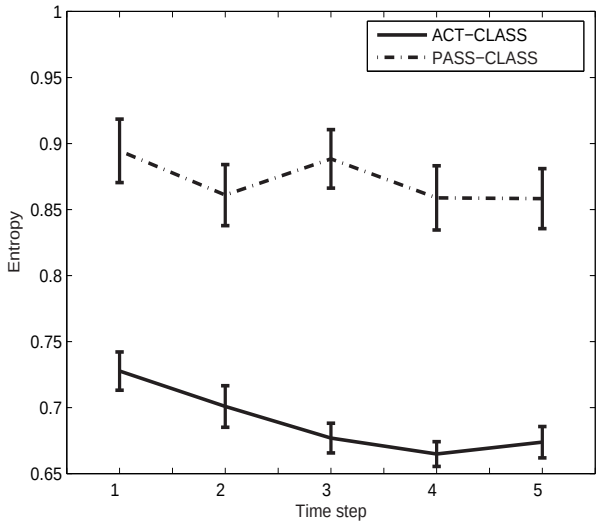


Figure 4.10: Mean entropy over time and corresponding standard errors for all best evolved instances of PAS-CLASS (dashed-dotted line) and ACT-CLASS (solid line).

The entropy of the posterior distribution is inversely related to the mutual information I (Shannon, 1948) between observations and classes: $I(C; O \mid i) =$

$H(C \mid i) - H(C \mid O, i)$. Therefore, we may also state that ACT-CLASS' controller maximises the mutual information between single observations and the classes over time.

4.6 Discussion

Below, we first discuss the use of information theory for studying sensory-motor coordination. Then we discuss our empirical findings in the light of the Markov assumption.

4.6.1 Information and Sensory-motor Coordination

At the beginning of the chapter, we defined sensory-motor coordination as the exploitation of the closed loop of motor outputs and sensory inputs in such a way that the performance on a particular task is optimised. This definition leads to different gradations of sensory-motor coordination related to how proficient an active model is in performing its task (think of active models at different stages in evolution).

Not all studies of sensory-motor coordination employ the same definition. An alternative view is that the gradation of sensory-motor coordination corresponds to the extent to which the actions influence the inputs (see, e.g., Klyubin, Polani, and Nehaniv, 2004).

Studies based on this different definition typically measure the mutual information between sensory inputs and actions. This measure can be used for (1) estimating the amount of internal processing of an active model (Thornton, 2005), (2) learning how to control actuators (Klyubin *et al.*, 2004), or (3) estimating what activity an active model is performing (te Boekhorst, Lungarella, and Pfeifer, 2003; Tarapore, Lungarella, and Gómez, 2006). However, mutual information between sensory inputs and actions per se does not do justice to the different contexts in which sensory-motor coordination is employed. Namely, it collapses this information to one single number, independently of the context. The definition of sensory-motor coordination and the manner in which we measure it should allow for these differences of models and tasks.

This is the case for the measure used in this chapter (the posterior distribution's entropy), since we can replace the variable C by an infinite number of other variables that might be relevant to other active models or other tasks. The mutual information between sensory inputs and other variables can be easily used as an analysis tool for active models that perform other state estimation tasks than classification. For example, for a task of estimating the distance to an object, we can replace C by a variable that represents the distance to the object, d_{obj}.

4.6.2 The Markov Assumption

In Chapter 2 we already encountered the Markov assumption: the future state only depends on the current state, observation, and action. It does not depend on observations and actions further in the past. In other words, the state description is

complete in the sense that it contains all variables of importance for the prediction of the future state.

This assumption is central to the algorithms employed by the probabilistic approach to active vision (Thrun, Burgard, and Fox, 2005), since it facilitates updating the belief state by allowing recursive updates. Without the assumption, the belief state would have to be updated on the basis of the entire history of observations and actions. The estimation of the posterior distributions involved is notoriously difficult, especially if the history can extend over many time steps.

The problem is that the Markov assumption is not always valid. Thrun *et al.* (2005) identify four factors that induce violations of the assumption. We repeat them here in our own words. (1) A part of the environmental dynamics has not been included in the state variables. (2) There are inaccuracies in the probabilistic models that lead to inaccurate probability distributions over the possible states. (3) There are approximation errors when using approximate representations of belief functions. (4) There are software variables that influence multiple controls. Thrun *et al.* (2005) add that in general, probabilistic algorithms are surprisingly robust to such violations. However, there are tasks in which a violation of the Markov assumption does pose a problem for the model's performance.

Exploiting the Violation of the Markov Assumption

Interestingly, ACT-CLASS seems to *exploit* the violation of the Markov assumption. It establishes dependencies between multiple actions and observations in order to enhance performance on the classification task (see Section 4.5).

In this subsection, we show the exploitation of the violation of the assumption for a model with a discrete observation set O and action set A. The active vision model has to select actions on the basis of its observations, so for the model O is the state space, as is the case in many reinforcement learning tasks (cf. Sutton and Barto, 1998). Consequentially, we have to show that $p(o_{i+1} \mid \mathbf{o}_i, \mathbf{a}_i) \neq p(o_{i+1} \mid o_i, a_i)$ in order to show that the task is non-Markovian[7]. To show that the active vision model exploits the non-Markovian nature of the task, we have to show that it uses the dependencies between the observations over multiple time steps to perform the task.

In what follows, we first explain shortly the main properties of the discrete model. For a detailed explanation the reader is referred to de Croon *et al.* (2005a). Then, we show that the task is non-Markovian. Finally, we explain how the model exploits the dependencies between the observations over multiple time steps.

Discrete Model

Figure 4.11 shows an overview of the discrete model. We note that the model has three modules, because it has a separate classifier and controller. The model still has a closed loop of inputs and actions, since it determines its gaze shifts on the basis of its observations. It therefore remains an active model of gaze control.

[7] As in Chapter 2 a bold letter represents a sequence until a certain time step, i.e., $\mathbf{o}_i = \langle o_1, o_2, \ldots, o_i \rangle$.

Module I extracts features in the same manner as ACT-CLASS, but it maps the extracted feature values to a set of discrete observations O. The classifier (module II) maps an observation to a class. In contrast to ACT-CLASS, the discrete model only classifies the image at the *last* time step of a run. At every time step during a run, the current observation ($\text{obs}_i \in O$) is mapped by a controller (module III) to an action ($a_i \in A$). The action set A is illustrated in Figure 4.12: the model can move to eight fixation locations around its current location. Both the controller and the classifier are tables of size $O \times A$, so the discrete model does not have a memory. The evolved model we discuss, had $|O| = 3$ and $|A| = 8$.

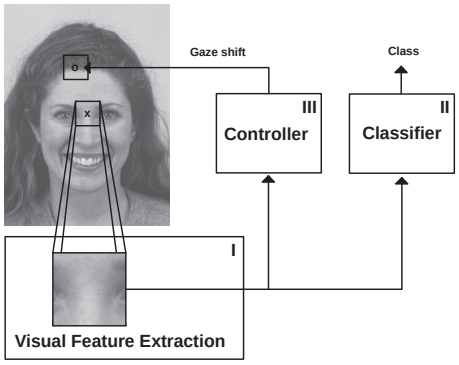


Figure 4.11: Overview of the active vision model. An 'x' marks the current fixation location. The three modules that constitute the model are illustrated by the boxes I, II, and III.

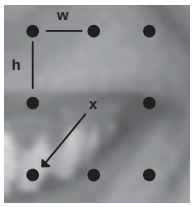


Figure 4.12: Grid indicating the locations to which the model can shift its gaze from the current fixation location (x).

Classification Task is Non-Markovian

As mentioned, for the active vision model the observation is the state. Therefore, the task is non-Markovian, if $p(o_{i+1} \mid \mathbf{o}_i, \mathbf{a}_i) \neq p(o_{i+1} \mid o_i, a_i)$. For the gaze control task this means that the probability distribution over the observations should not only be dependent on the last observation and last action. One example suffices to show this.

Let us look at Figure 4.13. It shows the fixation locations of the active vision model on the entire training set, with all different gaze paths exhibited by the model. The background image is included to give a rough impression of how the fixation locations relate to facial features. The active vision model encounters three possible observations. At the location indicated by an '$\times$' the probability distribution over these observations is $p(\text{obs}_1) = 0.050$, $p(\text{obs}_2) = 0.036$, and $p(\text{obs}_3) = 0.914$. At the location indicated by a '*', the probability distribution is $p(\text{obs}_1) = 0.807$, $p(\text{obs}_2) = 0.036$, and $p(\text{obs}_3) = 0.157$. Although the probability distributions at these fixation locations are different, one can arrive at the locations '*' and '$\times$' by perceiving obs_3 and taking an action of going to the top left: $p(o^{\times}_{i+1} \mid \text{obs}_3, a_{\text{top left}}) \neq p(o^{*}_{i+1} \mid \text{obs}_3, a_{\text{top left}})$. In other words, it is not possible to predict the probability distributions accurately on the basis of $o_i = \text{obs}_3$ and

$a_i = a_{\text{top left}}$ alone. On the contrary, if we have access to the history of all actions, we could deduce the current fixation location and therefore the exact probability distribution. Hence, $p(o_{i+1} \mid \mathbf{o}_i, \mathbf{a}_i) \neq p(o_{i+1} \mid o_i, a_i)$: the task is non-Markovian.

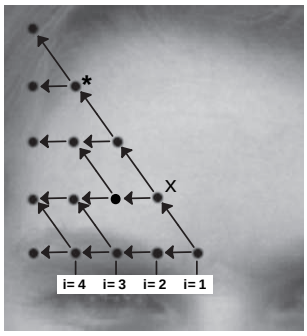


Figure 4.13: Fixation locations of the model on the whole training set, with all different gaze paths. Each fixation location is indicated with a dot, and each gaze shift with an arrow. The run starts at the bottom right location ($i = 1$).

Exploiting Dependencies over Multiple Time Steps

The active model exploits the dependencies of the observations over multiple time steps. Its general strategy is as follows. The first fixation location at the centre of the image is usually located at the eyebrows (Figure 4.13, bottom right, $i = 1$). Darker and thicker eyebrows result in obs_1 or obs_2, so that the model shifts its gaze to the left. For such images, the gaze keeps shifting to the left for the remaining three time steps, generally ending with obs_2 at the final time step. The classifier maps obs_2 to the male class. If the eyebrows are lighter, thinner, or not present at the first fixation location, the resulting observation is obs_3. Consequently, the model shifts its gaze to the top left, located on the forehead. This leads to a new observation of obs_3. Only if the model fixates the hair(line) of a person, the observation changes to obs_1. The controller maps obs_1 to a gaze shift to the left, into to the hair of the person, which leads once more to an observation of obs_1. This observation is mapped to the female class by the classifier. If the model does not reach the hairline within 4 time steps, the observation is still obs_3, which is associated with male images.

In other words, for male images the model verifies a disjunction of facial properties (dark eyebrows or a high hairline) and for female images a conjunction of facial properties (light eyebrows and a low hairline). The verification of the disjunction and conjunction implies the dependencies of the observations over a time span of 3 to 5 time steps.

Adaptive Active Vision Models and Non-Markovian Tasks

As mentioned above, it is notoriously hard to estimate the posterior over multiple time steps without making the Markov assumption. The number of observations

necessary for a reasonable estimate is usually too large. Therefore, one might wonder how it is possible for adaptive active vision models to make these estimates implicitly.

We suggest that the exploitation of observation over multiple time steps is mainly possible for the following two reasons. First, the strategy of the final adapted active model is based on a large number of runs performed during evolution. Second, the model's behaviour restricts the sequences of observations that it has to learn about.

One may argue that a third reason is the simplicity of this task, which involves only 3 different types of observations and 5 time steps. However, we feel that this is not a main reason for our finding, because other tasks that involve more time steps give similar results. For example, in Nolfi and Marocco (2002) and van Dartel *et al.* (2005) the evolved behaviour also exploits dependencies of observations, but over a larger number of time steps.

Of course, when employing a probabilistic active vision model, we can remodel the active vision process in order to make it Markovian. For instance, variables can be added to the state description to make it as complete as possible. In the study above, both the class and the fixation location could be included in the state description. However, for difficult problems we may not always know which are the best variables to add, and some tasks may be inherently non-Markovian. Our analysis above shows that adaptive active vision models form a promise for such problems.

4.7 Chapter Conclusion

In this chapter, we focused on RQ 2: *How does a memoryless adaptive gaze control model handle an image classification task?* From the empirical results, we may conclude that the adaptive active gaze control model, ACT-CLASS, uses sensory-motor coordination to maximise the information in its observations on the image class.

ACT-CLASS shifts its gaze to good classification locations on the basis of the class structure and the specific image structure. It uses its fixation location as an external memory to exploit dependencies between multiple observations and actions. This corresponds to exploiting the non-Markovian properties of its task.

The fact that we studied a *state estimation task* (classification) facilitated our analysis. In such a task it is clear what type of information the active vision model is attempting to extract from the visual scene: information on the state (in our case on the class of the image).

In the next chapter, we focus on a *control task*. Such a task is more difficult to analyse, since we do not know what information the active vision model gathers from the environment in order to perform its task as good as possible.

Chapter

5

Gaze Control and Information Gathering

The material in this chapter has not yet been published. Nonetheless, I would like to thank M. Suzuki and D. Floreano, with whom I collaborated on this research at the Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland.

In this chapter, we focus on RQ 3: *How does an adaptive gaze control model use its gaze shifts in a control task?*

The central assumption underlying almost all existing gaze control models is that the gaze shifts have as purpose to collect information about the environment. This assumption seems so natural that it is typically either omitted or mentioned as a triviality. If mentioned, any reader will skim over it without giving it much thought. For example, Lee and Yu (1999) state the following.

> *"When we 'look' (foveate) at a certain location in the visual scene, we direct our high-resolution fovea to analyze information in that location, taking a snap shot of the scene using our retina."*

When reading this statement, a sceptical reader may have his doubts concerning the taking of a "snap shot" of the scene. However, he will probably not have problems with the statement that "we direct our high-resolution fovea to analyze information in that location".

We conjecture that the gaze shifts of an adaptive gaze control model will also serve other purposes than information gathering from the environment. Intuitively, one can see that this might be the case, because information gathering is a means for achieving successful control, not a final goal.

Our conjecture is in line with findings on human vision. There is empirical evidence that humans use their gaze shifts for other purposes than collecting information from the environment. For example, human eye movements assist the process of mental visualisation (Laeng and Teodorescu, 2002), and support non-verbal social communication (e.g., Emery, 2000; Kampe *et al.*, 2001; Richardson *et al.*, 2006; Poel *et al.*, 2007).

Figure 5.1: View of the driving simulator. The car drives slightly left of the road centre. In the bottom right of the screen there is a tachometer, which indicates the working speed of the car engine.

The contradiction between our conjecture and the central assumption of other gaze control models incites us to investigate RQ 3 with a focus on information gathering. We aim to find a single example, in which gaze shifts are useful to the model's task but do not serve the gathering of information from the environment. Such an existence proof would suggest the potential existence and relevance of similar gaze shifts in natural systems.

To answer RQ 3, we analyse an active vision model that is adapted to the dynamic control task of simulated car-driving. We chose this task, since it is a real-time control task in which gaze shifts are commonplace and play an important role (Land, 1992; Land and Lee, 1994; Wilkie and Wann, 2003; Wann and Wilkie, 2004). The active vision model in this chapter is a controllable car, which has to drive and control its gaze in the same time. The model is an instantiation of ACT-FRAME, and is referred to as ACT-DRIVING. Its controller consists of two separate modules. The first module maps the visual inputs to car commands, which determine the speed and direction of the car (the *car controller*). The second module maps visual inputs to gaze shifts (the *eye controller*). We employ the simulator used in Floreano *et al.* (2004). It generates a virtual environment, which contains ACT-DRIVING and a driving track. In our experiments, we extend the original simulator with obstacles. Figure 5.1 shows a view of the extended driving simulator. ACT-DRIVING has to perform the task of following the road and avoiding the obstacles.

The remainder of this chapter is outlined as follows. In Section 5.1, we introduce ACT-DRIVING. Then, in Section 5.2 we explain our experimental setup. Our experiment consists of two phases. In the first phase, ACT-DRIVING is evolved to perform the driving task. For our investigation, it is necessary to obtain at least one successful instance of ACT-DRIVING. We show the outcome of the evolution in Section 5.3. In the second phase, we analyse the best evolved instance of ACT-DRIVING in order to determine the functions of its gaze shifts. We analyse the best evolved instance of ACT-DRIVING in Section 5.4. Our analysis aims at demonstrating that ACT-DRIVING does *not* only make gaze shifts to gather information about the environment. In Section 5.5 we discuss the implications of our findings. Finally, we draw the chapter conclusion in Section 5.6.

5.1 ACT-DRIVING

ACT-DRIVING has to perform the driving task on the basis of local image samples. It has to extract visual inputs from these samples, and then use these inputs to control both the car and the gaze shifts. In this section, we first explain ACT-DRIVING's visual feature extraction and then its controller. Finally, we mention the adaptable model parameters.

5.1.1 Visual Feature Extraction

ACT-DRIVING takes its inputs from a square gaze window in the rendered view (see Figure 5.1). Its gaze window has a size of $s \times s = 50 \times 50$ pixels, while the total rendered view has a size of 600×400 pixels. This makes the task quite challenging for ACT-DRIVING, since the window covers a small portion of the visual field. Because the simulated driving task does not involve natural images, it suffices to extract straightforward features from the raw pixel values in the gaze window. We use the same feature extraction as in Floreano *et al.* (2004). The pixel values in the window are first transformed to gray values defined on the interval from -1 (black) to +1 (white). Then, the window is subdivided into $m \times m = 5 \times 5$ input cells that form a matrix, resulting in 25 visual inputs. The inputs are extracted from the cells either by (1) taking the average gray value of all pixels in the cell, or by (2) taking the gray value of the centre pixel in the cell.

5.1.2 Controller

ACT-DRIVING has a neural network controller. It consists of two separate neural networks that form two separate modules, as illustrated by Figure 5.2. One module determines the car commands (the car controller) and the other module determines the gaze commands (the eye controller).

Both modules receive the same inputs. There are $m^2 = 25$ visual inputs, as explained above. In addition, ACT-DRIVING receives four *proprioceptive* inputs, i.e., inputs that provide information on ACT-DRIVING itself: the first represents its speed, and the remaining three are equal to the outputs of the eye controller[1].

As in previous studies (e.g., Floreano *et al.*, 2004), the perception-action cycle is set to 50 ms. Below, we first discuss the type of neural network employed for both controller modules. Then we explain the car controller and subsequently the eye controller.

Continuous Time Recurrent Neural Networks

Preliminary experiments showed that feedforward neural networks are not successful at controlling ACT-DRIVING. Therefore, we chose the commonly employed

[1] We do not include a recurrent connection from the eye outputs to the input layer in Figure 5.2, since this would mean that the two layers are completely connected. However, we note that the proprioception of the eye outputs does constitute a mechanism of recurrency.

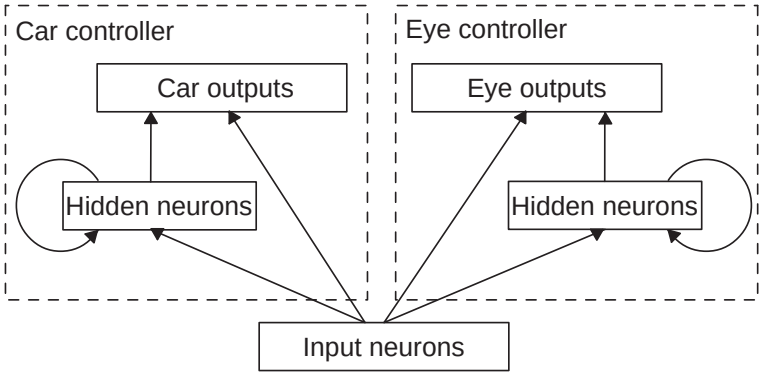


Figure 5.2: Controller of ACT-DRIVING. The controller has a modular structure. The inputs are forwarded to two separate neural networks, both with recurrent hidden neurons. The car controller determines the car commands (forward / backward and steering), and the eye controller the eye commands (gaze location and feature extraction method). A box represents a collection of neurons, an arrow indicates that all neurons of two boxes are connected in the direction of the arrow.

Continuous Time Recurrent Neural Networks (CTRNNs) for the controller modules. CTRNNs have recurrent connections and the neural activity of each neuron has its own inertia. As a consequence, ACT-DRIVING's controller has an internal state which can exhibit various dynamics (Funahashi and Nakamura, 1993; Beer, 1995; Beer, 2003; de Croon, Nolfi, and Postma, 2006a). The activation of a neuron j in a CTRNN is determined by the following differential equation.

$$\tau_j h'_j = -h_j + \sum_{k=1}^{q} w_{kj} \text{tanh}(g_k(h_k + \theta_k)) + u_j \tag{5.1}$$

In the equation, we have: h_j as the activation potential of neuron j, τ_j as its *time constant*, q as the number of neurons that provide inputs to the neuron, and w_{kj} as the corresponding weights. Following Floreano *et al.* (2004), the weights are constrained to $w_{kj} \in [-w_{\text{max}}, w_{\text{max}}] = [-4, 4]$. Furthermore, tanh is the activation function: $a(z) = \text{tanh}(z) = 1 - \frac{2}{1+e^{2z}}$. As a consequence, each neural activation is in the interval $\langle -1, 1 \rangle$. Finally, g_k is the *gain* of neuron k, θ_k its bias, and u_j is the sensor input to neuron j (e.g., a visual input). To simulate the network, we use an Euler approximation to the differential equation, with a simulation step of $\Delta i = 0.2$ (corresponding to 10 ms). Each module has 5 hidden neurons.

Car Controller

The *car controller* maps the inputs to two outputs. The first output, out_{c1}, encodes for the forward command ($\text{out}_{c1} > 0$) or the backward command ($\text{out}_{c1} < 0$) to the car, where 0 represents no acceleration. The second output, out_{c2}, represents the steering direction of the car: -1 is steering to the right, 0 is straightforward, and 1 is steering to the left.

Eye Controller

The *eye controller* maps the inputs to three outputs. The first two outputs of the eye controller, out_{e1} and out_{e2}, encode for the *absolute coordinate* of the gaze location in the rendered view of the simulator. By selecting different gaze locations, ACT-DRIVING can shift its gaze within the view. The formula for decoding the outputs to the gaze location is: $(x, y) = ([549 \times ((\text{out}_{e1} + 1)/2)], [349 \times ((\text{out}_{e2} + 1)/2)])$, where $[\cdot]$ is the round-function[2]. As a consequence, an x- or y-output of -1 stands for the left or top of the screen, respectively. To prevent ACT-DRIVING from making implausibly large gaze shifts, we set a maximum $d_{\text{max}} = 250$ to the horizontal and vertical shift in the image. The third output, out_{e3}, determines the feature extraction method as follows: ACT-DRIVING takes the average gray value from all input cells if $\text{sign}(\text{out}_{e3}) = 1$ and the centre gray value if $\text{sign}(\text{out}_{e3}) = -1$.

5.1.3 Adaptable Model Parameters

ACT-DRIVING has three types of adaptable model parameters, all related to the CTRNN: (1) the time constants τ_j, (2) the gains g_j, and (3) the weights w_{kj}.

5.2 Experimental Setup

In this section, we describe the experimental setup for the simulated driving task. We explain the task specifics and the evolutionary algorithm used to optimise ACT-DRIVING. In addition, we specify the analytical manipulations used to investigate the function of ACT-DRIVING's gaze shifts.

5.2.1 Simulated Driving Task

In the simulated driving task, ACT-DRIVING has to drive over a track as quickly as possible, while avoiding obstacles on the road. Below, we first discuss the simulated environment. Then we explain how we evaluate ACT-DRIVING's performance in the simulator by letting it drive multiple *runs*. For each run, a different track is generated with different positions of the obstacles. We explain how a track is generated, and how the obstacles are placed on the track.

Simulated Environment

Figure 5.1 shows an example view of the simulator. As mentioned above, it has a resolution of 600×400 pixels. Each 2-D view is generated from the 3-D simulation environment, with as reference point the centre of the car. The view is in the direction of the car body. The car itself is not visible in the view; it is 2.3 metres wide and 5.6 metres long. The setting of the landscape in the simulator is the same throughout evolution and testing: a flat, grassy landscape, with a gray road demarcated by white lines. The obstacles on the road are always white. We note that

[2] $[x] = \lfloor x \rfloor$, if $x - \lfloor x \rfloor < 0.5$; and $[x] = \lceil x \rceil$, if $x - \lfloor x \rfloor \geq 0.5$.

this setting reflects that we are interested in the gaze shift strategy of ACT-DRIVING, and not in a realistic driving system (cf. Dickmanns, Mysliwetz, and Christians, 1990; Pomerleau, 1995; Baluja and Pomerleau, 1997; Thrun *et al.*, 2006). The environment contains multiple elements at different positions in the visual field that are important to ACT-DRIVING's task (obstacles, road, and road line). Therefore, we expect ACT-DRIVING to shift its gaze to all these elements.

Evaluation of Driving Performance

To evaluate ACT-DRIVING's performance, we let it drive multiple runs in the simulator. At the start of each run, we initialise ACT-DRIVING by placing it at the beginning of the driving track. Its initial orientation is at an angle randomly selected from the interval $[-90°, 90°]$, where $0°$ means that the gaze direction of ACT-DRIVING is aligned with the direction of the road. The initial gaze location is set to the centre of the 600×400 view. After this initialisation, ACT-DRIVING has to drive as far as possible on the track within a limited time interval. The run ends if it (1) goes off-track, (2) hits an obstacle, or (3) exceeds a driving time of t time steps.

Track Generation

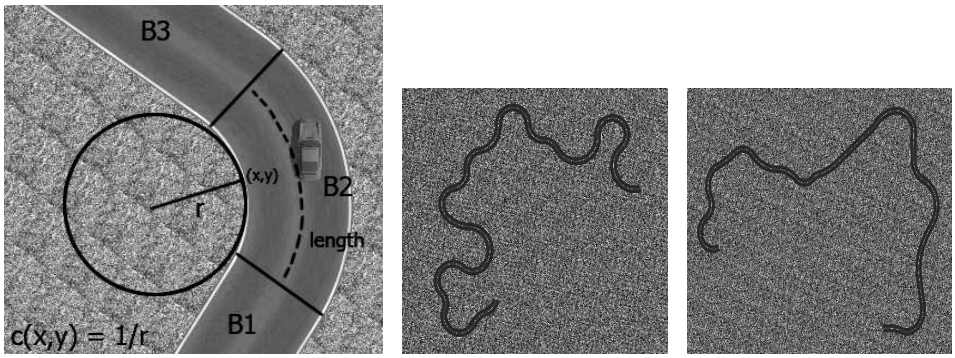
For each run, we generate a different track, which consists of a sequence of 50 *track blocks*. Figure 5.3(a) shows three track blocks, B1, B2, and B3. Every track block is generated on the basis of two variables, the length l and the curvature c. The width of each track block is constant, and amounts to 10 metres. In the figure, the length is indicated by a dashed line. The curvature c of a point (x, y) on the road is defined as the reciprocal of the radius r of the inner line's *osculating circle*: $c(x, y) = \frac{1}{r}$ in the figure. The osculating circle is defined as the circle that touches the road line at (x, y). If the track has a high curvature, the osculating circle is small and so is its radius. If the track block is straight, the radius of the osculating circle is infinitely large, and the curvature is 0.

To generate a track block, we take independent samples from two distributions: a length distribution and a curvature distribution. These distributions represent the probabilities that a track block has a specific length or curvature, respectively. Figure 5.3(b) shows a track with high curvatures. Figure 5.3(c) shows a track with low curvatures. The length distribution and curvature distribution used in the experiments are shown in Figure 5.4.

After determining the length and curvature of the track block, we randomly select a direction for the curve: it goes either left or right. To minimize the probability of self-intersections, not more than two subsequent track blocks can curve in the same direction. In the rare cases where a generated track block does intersect with the already generated part of the track, it is deleted and a new track block is generated.

Obstacles

Once the track is generated, 11 obstacles (with a floor surface of 1.5×1.5 metre) are placed on it as follows. The first two obstacles are placed near the beginning of



(a) Track block generation (b) High-curvature track (c) Low-curvature track

Figure 5.3: **(a)** Three track blocks, B1, B2, and B3. See the text for a detailed explanation. **(b)** A track generated by sampling from a curvature distribution with high curvatures. **(c)** A track generated by sampling from a curvature distribution that is skewed towards low curvatures.

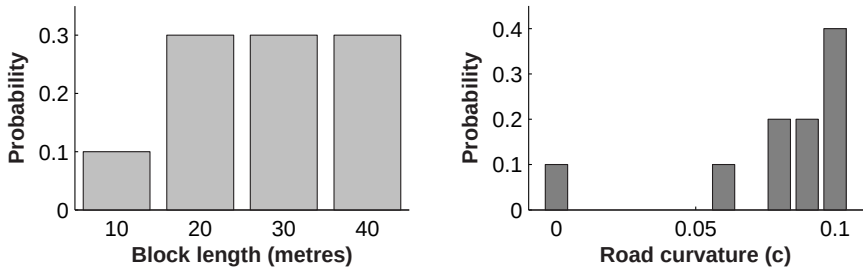


Figure 5.4: Probability distributions of track block lengths (left) and curvatures (right). For track generation, we take independent samples from these distributions.

the track and on opposite sides of the road. The first obstacle is placed randomly between 3% and 5% of the track length as measured from the starting point, on either the left or the right side. The second obstacle is randomly placed between 8% and 10% of the track length, on the opposite side of the first obstacle. We refer to an instance of ACT-DRIVING as *successful* if it succeeds in driving past the first 10% of the track, since then we know that it is able to drive *and* avoid obstacles. The remaining 9 obstacles are randomly placed on the rest of the track, with a minimal separation of 5% of the track length. All obstacles are placed either on the left or the right side of the track (i.e., not in the middle of the road).

5.2.2 Evolutionary Algorithm

We use a λ, μ-evolutionary algorithm (Bäck, 1996) to optimise the adaptable parameters of ACT-DRIVING. Each generation consists of a population of $\lambda = 100$

instances of ACT-DRIVING.

The genome of ACT-DRIVING has three parts, consisting of double values in the interval $[-1, 1]$. The first part of the genome encodes for the time constants τ_j associated with each neuron j. The second part of the genome encodes for their gains g_j. The third part of the genome encodes for the weights w_{kj} of the neural network. During decoding the parameters are cast to the following intervals. The time constants are decoded so that $1/\tau_j$ is in the interval $[0, 1]$, the gains are decoded to the interval $[0.01, 10]$, and the weights to the interval $[-w_{\text{max}}, w_{\text{max}}] = [-4, 4]$. The values of the genes in the genome determine the behaviour of ACT-DRIVING.

The $\mu = 20$ best instances of ACT-DRIVING are selected to form the next generation. The fitness function used to evaluate an instance is:

$$f = \frac{\sum_{r=1}^{n} l_r / l_{\text{track}}(r)}{n}, \tag{5.2}$$

where l_r is the length of the track covered by ACT-DRIVING during run r, and $l_{\text{track}}(r)$ is the total length of the track. We remark that the total track length depends on the run, since we always generate a new track for each run. Each run maximally lasts $t = 700$ time steps (35 seconds in simulation).

f is the average over $n = 5$ runs of the proportion of the track covered by ACT-DRIVING, and therefore $f \in [0, 1]$. The motivation behind the fitness function is that it is uncomplicated and yet stimulates fast driving, obstacle avoidance, and staying on the track. This is enforced by the stopping criteria explained in Subsection 5.2.1.

The best $e = 3$ instances of ACT-DRIVING are copied to the new generation. The remaining $\lambda - e$ instances are created by applying crossover and mutations to the μ best instances. One point crossover between two selected instances of ACT-DRIVING is applied with a probability of $p_{\text{co}} = 0.05$. After a possible crossover, the offspring's genes are mutated with a probability of $p_{\text{mut}} = 0.04$. A mutation of a gene gives it a random value in its corresponding gene interval. Evolution is continued for $g = 150$ generations. To obtain our experimental results, we perform $v = 5$ independent evolutions.

5.2.3 Analytical Manipulations

In Section 5.4, we analyse the best evolved instance of ACT-DRIVING. The goal of the analysis is to find out the functions of the different types of gaze shifts. For each type of gaze shift, we analyse the function by manipulating ACT-DRIVING.

Throughout our analysis, we mainly employ three different analytical manipulations: (1) disabling ACT-DRIVING's gaze shifts at certain time steps or in certain situations, (2) fixing a subset of the controller's neural activations, and (3) replacing ACT-DRIVING's visual inputs. We discuss these three manipulations below. In our discussion we emphasise the third manipulation, since it is essential to verifying our conjecture on the gathering of information.

(1) Disabling Gaze Shifts

The first manipulation is to disable ACT-DRIVING's gaze shifts. The goal of the manipulation is to find out the contribution of ACT-DRIVING's gaze shifts to the fitness. For example, we may disable ACT-DRIVING's gaze shifts from time steps $i = 100$ to $i = 200$. If there is no difference in fitness between ACT-DRIVING when it is free to move its gaze and when it has its gaze shifts disabled, then the gaze shifts do not contribute to the fitness during these time steps.

(2) Fixing Neural Activations

The second manipulation is to fix a subset of ACT-DRIVING's neural activations. The goal of the manipulation is to assess the influence of certain neural activations on ACT-DRIVING's behaviour or fitness. For example, to test if ACT-DRIVING uses the proprioceptive input that represents the car speed, we may fix the activation of this proprioceptive input to its initial value. If the manipulation has no effect on ACT-DRIVING's fitness, then the proprioceptive input serves no function in its behaviour.

(3) Replacing Visual Inputs

The third and most important manipulation is to replace ACT-DRIVING's visual inputs by plausible but uninformative inputs. The goal of the manipulation is to determine whether ACT-DRIVING gathers information from the environment in a certain situation.

For example, ACT-DRIVING may sometimes shift its gaze to the bottom right part of the rendered view. We may suspect that it does not extract information from its visual inputs from this part of the view, because it usually contains the grass texture. Then this can be tested as follows. We first run ACT-DRIVING on various tracks, without manipulating it. During these normal runs, we store the visual inputs when it gazes at the bottom right of the screen, annotating the inputs with the corresponding gaze location. Subsequently, we run ACT-DRIVING again on various tracks. However, this time we manipulate its visual inputs when it gazes at the bottom right of the view: instead of providing ACT-DRIVING with the visual inputs from the current rendered view, we provide it with stored visual inputs. To this end, the $k = 10$ stored inputs are determined, of which the corresponding gaze locations are closest to ACT-DRIVING's current gaze location[3]. Then, one of these inputs is selected at random.

As a consequence, ACT-DRIVING receives visual inputs that are plausible for the selected gaze location. But, importantly, since they have been gathered on a different track and in a different run, *these visual inputs do not convey any information on the current state of the road*. The stored input that replaces the actual input may have been gathered in a curve to the left, while ACT-DRIVING is now in a curve to the right. In this case, if ACT-DRIVING extracts information on the curve direction

[3]It should be noted that we do not select the $k = 10$ stored inputs that are closest in the input space. Such a method would not lead to a valid test.

from the visual inputs, replacing the input should degrade its performance. Of course, there is a small possibility that a stored input is equal to the actual input. In that case, replacing the input does not influence ACT-DRIVING's performance. Since it is highly unlikely that all of the replacing inputs are equal to the actual inputs, the third manipulation can show whether ACT-DRIVING extracts information from its visual inputs. If it does, its performance should degrade when replacing the current inputs with the stored inputs.

We emphasise that in our analysis we search for gaze shifts that *do* contribute to ACT-DRIVING's performance, but that do *not* serve to gather information from the environment.

5.3 Evolution

We need at least one instance of ACT-DRIVING that is successful in the driving task, i.e., an instance with a fitness higher than 0.10 (10% of the track). In this section, we present the outcome of evolution and determine the most successful instance of ACT-DRIVING.

Figure 5.5 shows the fitness of the best instances of ACT-DRIVING over the different generations of the evolution. During evolution, we test each instance of ACT-DRIVING for $n = 5$ runs to obtain an impression of its fitness. To determine the best instance of ACT-DRIVING of all different evolutionary runs in a reliable manner, we need a larger number of runs. First the best instance of each evolution is selected as follows. All instances that were the best of their generation are tested for $n = 100$ runs, and the instance of ACT-DRIVING with the highest average fitness is selected. Then, this best instance is tested for $n = 1000$ runs. For each instance of ACT-DRIVING, we register seven different metrics that together provide an overview of its performance on the driving task. The metrics and their values for the best instances of ACT-DRIVING of all evolutionary runs are shown in Table 5.1. It shows the following metrics: the maximum fitness ('max'), average fitness ('mean'), standard deviation ('st. dev.'), standard error of the mean ('st.e.m.'), the proportion of runs ending in a crash ('crash'), the proportion of runs ending in the car going off-track ('off-track'), and the proportion of runs in which ACT-DRIVING runs out of time ('time').

Table 5.1: Test performance on $n = 1000$ runs of the best evolved instance of ACT-DRIVING per evolution.

	max	mean	st.dev.	st.e.m.	crash	off-track	time
Evolution 0	0.671	**0.174**	0.111	0.004	0.623	0.085	0.292
Evolution 1	0.122	**0.015**	0.011	0.000	0.005	0.995	0.000
Evolution 2	0.222	**0.016**	0.014	0.000	0.001	0.999	0.000
Evolution 3	0.119	**0.019**	0.017	0.001	0.015	0.985	0.000
Evolution 4	0.661	**0.152**	0.073	0.002	0.454	0.105	0.441

From Figure 5.5 and Table 5.1 we may make three main observations. First, only two out of the five evolutionary runs succeed in finding a solution that surpasses

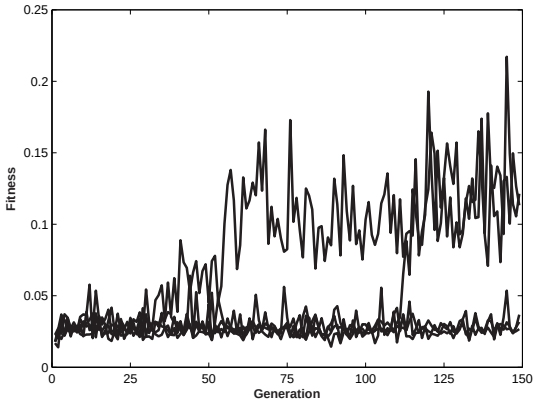


Figure 5.5: Fitness of the best instance of ACT-DRIVING of each generation for the five evolutionary runs.

0.10 (see column 'mean'), indicating that the instance of ACT-DRIVING successfully avoids the first two obstacles. The *evolvability* for the simulated driving task seems to be low: only a few runs result in successful instances of ACT-DRIVING.

Second, the average fitness values themselves appear to be rather low, with as maximum 0.174 (17.4% of the track). The reason for this lies in the long track length of 50 track blocks. We selected this long track length on purpose, so that we would be sure of continued evolutionary pressure on faster and better driving. The reader can verify that the track is rather long by looking at the example run shown in Figure 5.6. The figure shows the position of the car over time (dashed line), annotated with the time steps ($\Delta i = 25$). The road lines are represented by the solid lines, obstacles by (small) squares. We indicate the car at the last time step ($i = 700$, i.e., 35 simulated seconds) by a rectangle. The left part shows the entire track, the right part zooms in on the part covered by the car. We verified whether it is possible to finish the entire track by performing ten runs of 35 seconds with the simulated car ourselves. We obtained an average fitness of $\sim 30\%$. This is slightly faster than ACT-DRIVING, but we were able to see the entire screen at each time step. Therefore, the average fitness of 17.4% seems reasonable.

Third, Table 5.1 shows that evolution number 0 resulted in the best instance of ACT-DRIVING. It has the highest mean and maximum performances, and significantly outperforms the best instance of ACT-DRIVING of evolution number 4 ($p < 0.05$, randomisation test; Cohen, 1995, see also the standard error of the mean, 'st.e.m.'). The main reason for the performance difference seems to be that the instance of evolution 0 drives faster than the instance of evolution 4. The instance of evolution 0 runs out of time less often than the instance of evolution 4, since it crashes more often. The best instance of ACT-DRIVING of evolution 0 will be the subject of our analysis in the next section.

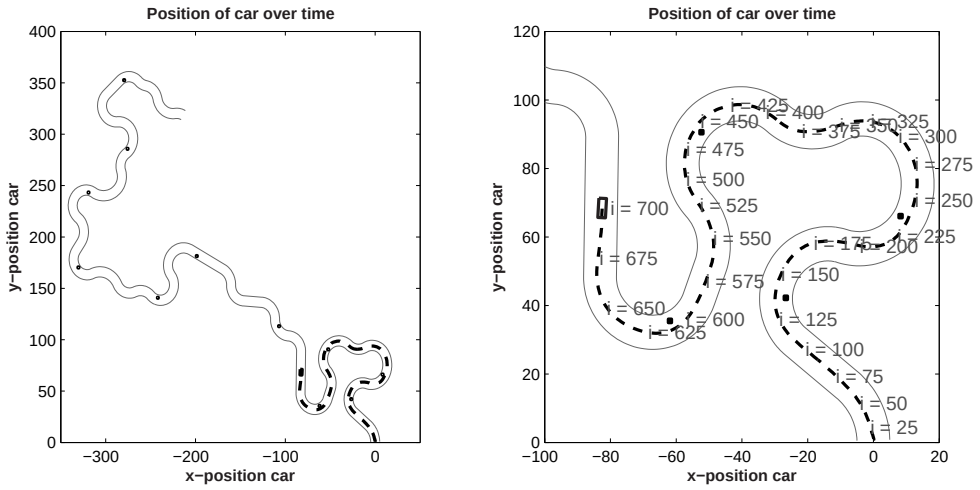


Figure 5.6: Example run of ACT-DRIVING. **Left:** Overview of the entire track. **Right:** Overview of the part covered by the car. We show the position of the car over time (dashed line), annotated with the time steps ($\Delta i = 25$). The road lines are represented by the solid lines, obstacles by (small) squares. We indicate the car at the last time step ($i = 700$, i.e., 35 simulated seconds) by a rectangle.

5.4 Analysis

Below, we perform an analysis in order to find out whether and how the most successful instance of ACT-DRIVING[4] uses its gaze shifts to improve its performance on the task.

We start the analysis by plotting ACT-DRIVING's gaze shifts over time. Figure 5.7 shows the x (solid) and the y (dashed) coordinate of ACT-DRIVING's gaze location over time for one of the testing runs[5]. It makes gaze shifts during driving. We test *whether* these movements help to improve the performance by employing the first manipulation. We disable the gaze shifts from the beginning of the run (time steps $i \geq 0$). Table 5.2 provides the performance of ACT-DRIVING under different testing conditions that are used for the analysis. The columns represent the same performance criteria as in Table 5.1, complemented by one column ('sign.') that indicates whether the average performance is significantly different from the normal driving condition (randomisation test, $p < 0.05$; Cohen, 1995). For now, the first two rows of the table are important. The first row shows the performance of ACT-DRIVING under normal conditions. The second row shows the performance when ACT-DRIVING is not allowed to make any gaze shifts (see the row 'No shifts'). Comparing the first two rows reveals that entirely disabling ACT-DRIVING's gaze shifts seriously deteriorates its driving behaviour: the average fitness is reduced to 0.009. The remaining rows will be discussed in subsections 5.4.1 to 5.4.3.

[4]From here on we abbreviate 'the most successful instance of ACT-DRIVING' to 'ACT-DRIVING'.

[5]We remark that the gaze shifts are not related to the run in Figure 5.6.

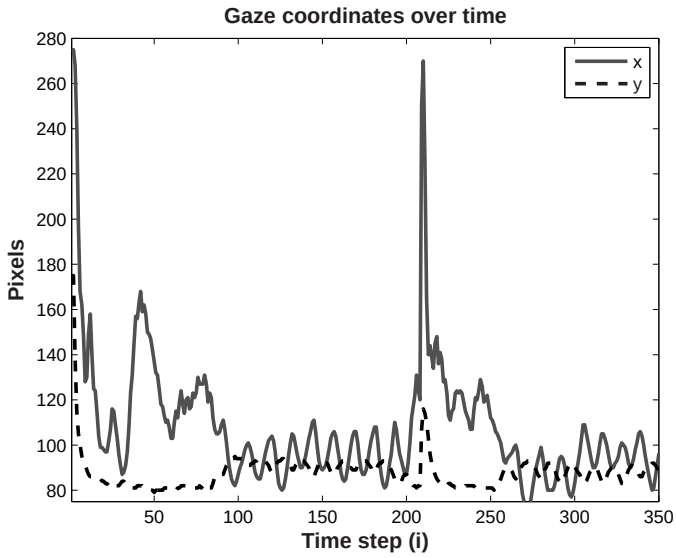


Figure 5.7: Gaze shifts during a run on a curved track. The plot shows the x-coordinate (solid) and y-coordinate (dashed) of the gaze location over time, in pixels. The coordinates represent the top left of the gaze window.

Table 5.2: Test performance of ACT-DRIVING on $n = 1000$ runs, under different conditions. The last column indicates whether the performance difference with the normal condition (top row) is statistically significant (✓) or not (-).

	max	mean	st.dev.	st.e.m.	crash	off-track	time	sign.
Normal	0.671	**0.174**	0.111	0.004	0.623	0.085	0.292	-
No shifts	0.142	**0.009**	0.011	0.000	0.001	0.999	0.000	✓
No shifts $i > 100$	0.180	**0.055**	0.020	0.001	0.105	0.894	0.001	✓
No eye reflex	0.874	**0.162**	0.105	0.003	0.538	0.214	0.248	✓
Replace inputs	0.739	**0.173**	0.110	0.003	0.606	0.082	0.312	-
No proprioc.	0.817	**0.173**	0.111	0.004	0.587	0.091	0.322	-

From this result, we infer that ACT-DRIVING uses its gaze shifts to achieve the registered performance on the task. Below, we show that the gaze shifts have three functions: (1) to find relevant features in the environment, (2) to keep relevant features in sight, and (3) to avoid disruptive visual inputs. The third function does not involve the gathering of information from the environment.

5.4.1 Find Relevant Features

The first function of the gaze shifts is to find relevant features in the environment. As explained in Subsection 5.2.1, ACT-DRIVING's gaze location is initialised to the centre of the view. This gaze location is not suited for the driving task, since it is

located at the border between the sky and the ground. It would be difficult for ACT-DRIVING to extract information from this location on where the road is going. An informative feature of the environment on where the road is going, is the road line. ACT-DRIVING's first action is to find this feature: it shifts its gaze from the centre location gradually to the bottom left until it finds the road line. We can see this in Figure 5.7. During the first 20 time steps, the x-coordinate of the gaze location decreases from 270 to 95 pixels, and the y-coordinate from 175 to 85 pixels. In the bottom left part of the view, the road line is thicker than further upward the view. ACT-DRIVING probably prefers the left part of the view over the right part, since the left line is not (partially) occluded by the tachometer.

5.4.2 Keep Relevant Features in Sight

The second function of the gaze shifts is to keep relevant features of the environment in sight. Below, we first demonstrate that the gaze shifts contribute to the fitness, even after ACT-DRIVING has located the road line. Then we show that ACT-DRIVING makes oscillatory movements with both its car body and its gaze. Finally, we explain how the gaze shifts augment the frequency with which relevant features of the environment are sampled, and why this is useful for the driving task.

Contribution to Performance

Figure 5.7 shows that ACT-DRIVING moves its gaze during the entire run. To test whether ACT-DRIVING's gaze shifts still improve its performance after it has found the road line, we disable the gaze shifts after time step $i = 100$. This gives ACT-DRIVING ample time to move its gaze location towards the road line, even if it starts with the maximum rotation of the car body. Table 5.2 shows the result of this test in the third row ('No shifts $i > 100$'). Disabling the gaze shifts after $i = 100$, results in an average fitness of 0.055. The gaze shifts after $i = 100$ contribute considerably to the performance.

Oscillatory Behaviour

How do the gaze shifts improve ACT-DRIVING's performance after the start of the run? To answer this question, we can make a key observation from Figure 5.7: the gaze shifts of ACT-DRIVING are oscillating for $i \in [100, 200]$ and $i \in [260, 350]$.

During the run ACT-DRIVING's car body also has an oscillatory behaviour. Figure 5.8 illustrates the position of ACT-DRIVING on a straight track, when it is allowed to make gaze shifts (left) and when it is not (right). The road is represented by the straight vertical lines at $x = -5$ and $x = 5$. The trajectory of the car is annotated by the corresponding time steps. In both plots, the annotations are $\Delta i = 25$ time steps apart[6].

[6]We note that on a straight track ACT-DRIVING drives in the centre of the road (slightly to the right). On a curved track, ACT-DRIVING drives slightly to the left of the centre (see Subsection 5.4.3).

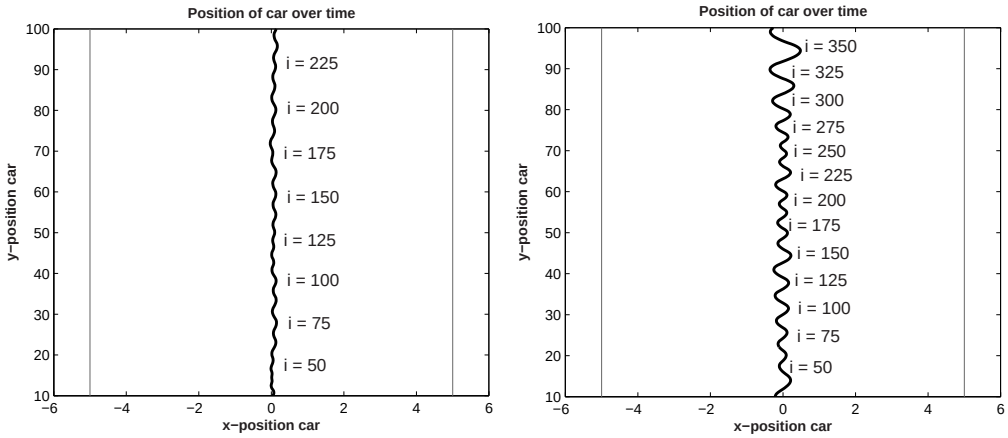


Figure 5.8: Position of the car on a straight track, when it is allowed to make gaze shifts, **(left)**, and when it is not, **(right)**. The road is represented by the straight vertical lines at $x = -5$ and $x = 5$. The trajectory of the car is annotated by the corresponding time steps. In both plots, the annotations are $\Delta i = 25$ time steps apart. We show the trajectory with solid lines to render the oscillations more visible.

Cause of Oscillatory Behaviour

How do the oscillatory movements of the car body originate? In animal locomotion, oscillating activations play an important role and are often generated by central pattern generators, CPGs (cf. Buchli, Righetti, and Ijspeert, 2006; Ijspeert and Crespi, 2007). Since CTRNNs are capable of complex internal dynamics (Funahashi and Nakamura, 1993; Beer, 1995), it may be possible that ACT-DRIVING uses the internal state of its controller modules to create an artificial CPG. If the oscillatory activations are caused by the internal state, then they should continue if we fix the inputs to the controller modules (second manipulation). However, when we fix the inputs, the oscillatory activations cease and ACT-DRIVING goes off-track. Figure 5.9 shows this for a run on a straight track, in which we fix the inputs at time step $i = 170$.

Therefore, the oscillatory behaviour of the car body has to arise as an effect of the coupling between the controller modules and the environment. We will explain the oscillatory behaviour by studying the relation between the visual inputs and the outputs of the car controller. To this end we employ the *correlation* between two variables over time, a and b.

$$\text{corr}(a, b) = \frac{\text{cov}(a, b)}{\sqrt{\text{var}(a)\text{var}(b)}} \tag{5.3}$$

$$\text{cov}(a, b) = E[(a - \bar{a})(b - \bar{b})] \tag{5.4}$$

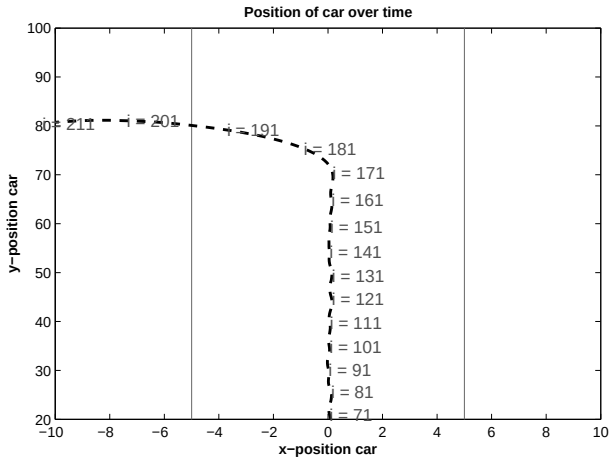


Figure 5.9: Position of car on straight track when fixing its visual inputs. The road is represented by the straight vertical lines at $x = -5$ and $x = 5$. The trajectory of the car is illustrated by the dashed line, and is annotated by the corresponding time steps. These annotations are $\Delta i = 10$ time steps apart. ACT-DRIVING's visual inputs are fixed at $i = 170$.

$$\text{var}(a) = E[(a - \bar{a})(a - \bar{a})] \tag{5.5}$$

We computed the correlation measure between activations of the output neurons and the visual input neurons, during a run of $t = 1000$ time steps. Figure 5.10 shows the outcome for the steering output of the car controller (left) and the output that encodes the x-position, out_{e1}, of the eye controller (right). The correlation values are shown at the corresponding positions of the visual sensors in the gaze window. The gray values encode for the correlation values. The values are scaled so that pure black represents maximal negative correlation, and pure white represents maximal positive correlation. The gray value of the figure's background corresponds to the value for no correlation. We interpret the correlation values shown in the figure as follows. A positive correlation value of a visual sensor means that the output activation increases, if the visual sensor is more active than usual (and decreases if the visual sensor is less active). A negative correlation means that the output decreases, if the visual sensor is more active than usual (and increases if the visual sensor is less active).

We can now explain the oscillatory movement of the car body with the help of Figure 5.10 and of Figure 5.11. The last figure shows the simulator view at different times during the oscillatory behaviour[7]. We have superposed a magnification of the visual input cells on each view, in order to better show the (subtle) differences in the visual inputs. In the left part of the figure the car body is more to the left, as can be seen by looking at the intersections of the road lines with the border of the screen.

[7]We note that the figure shows the behaviour while ACT-DRIVING is allowed to move its gaze.

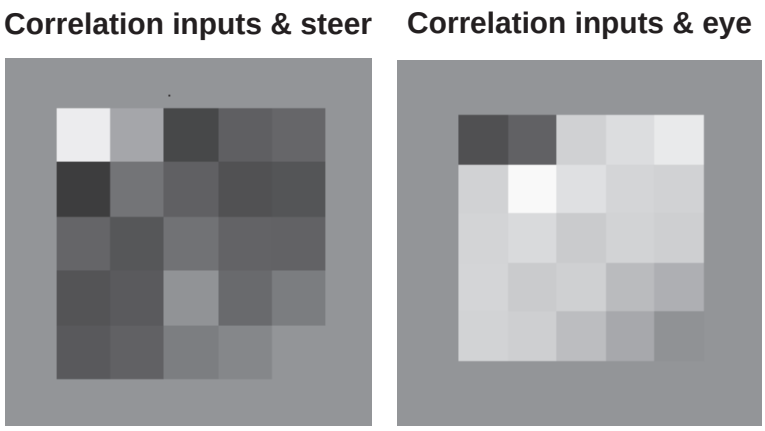


Figure 5.10: Correlation of the visual inputs with **(left)** the steering output of the car controller and **(right)** the output of the eye controller that encodes for the x-coordinate of the gaze location. The correlations have values in the interval $[-1, 1]$, where -1 is maximal negative correlation, 0 is no correlation, and 1 is maximal positive correlation. These values are represented with gray values, where pure black encodes for -1 and pure white for 1. We show the correlation values in the form of the input matrix. The gray value of the background surrounding the matrix represents a correlation value of 0.

The road line is located somewhat lower in the gaze window than in the right part of the figure. As a result, the input cells of the upper left triangle (except the top left one) are more active than on average. The correlation values in Figure 5.10 reveal that these cells are negatively correlated with the steering output. This means that the presence of the line in the window results in a (more) negative steering output: the car goes to the right. When the car goes to the right, the line moves up in the window towards the top left input cell (see right part of Figure 5.11). The top left cell is positively correlated with the steering output. If it is reached by the road line, it has a high activation while the other cells in the upper left triangle have a lower activation. This leads to a (more) positive steering output: the car goes back to the left.

Purposes of Oscillatory Behaviour

The oscillatory movements of the car body serve one main purpose: they allow ACT-DRIVING to make curves both to the right and to the left. When it is engaged in a curve to the right, the road line will move down in the gaze window, even if ACT-DRIVING is driving straightforward. Since the line will activate the diagonal input cells from the bottom left to the top right more often, ACT-DRIVING will steer more to the right. The inverse argument applies to curves to the left.

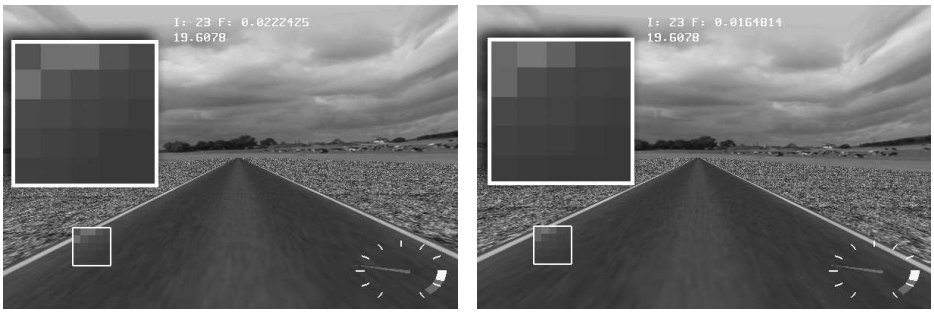


Figure 5.11: Two screen-shots of the simulator. Per screen-shot we include a magnification of the gaze window, to better show the activations of the visual inputs. The activations are in the interval $[-1, 1]$ and are scaled here from black to white. **Left:** As a reaction to the shown inputs, ACT-DRIVING steers to the right. **Right:** As a reaction to the shown inputs, ACT-DRIVING goes to the left. Because of these reactions, ACT-DRIVING 'wiggles' on a straight track.

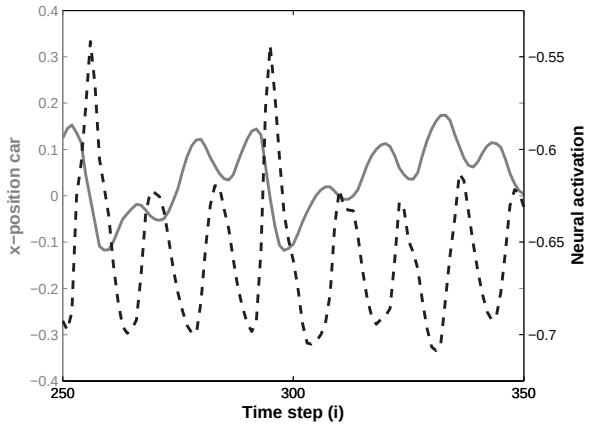


Figure 5.12: Plot of the neural activation of the x-output of the eye (dashed line) versus the x-position of the car on a straight track (solid line).

Function of Gaze Shifts

Now that we have an idea on how the oscillatory movements of the car body arise and why they are useful, we can focus on the function of the gaze shifts. The correlation values for the x-component of the gaze shift seem inverted from the ones of the steering output (see Figure 5.10). In Figure 5.12 we illustrate the resulting relation between the x-output of the eye (dashed line) and the horizontal position of the car (solid line) over time. We measure both these quantities for ACT-DRIVING on a straight track. The activation of the x-output of the eye, out_{e1}, should be interpreted as follows. If it is more negative, the gaze shifts more to the left, if it is more positive, the gaze shifts to the right. The figure shows that the gaze moves opposite of the car (and moves slightly before the car does).

As a result, the gaze shifts keep the line in sight. This increases ACT-DRIVING's speed. If we prevent ACT-DRIVING from making gaze shifts, it loses the road line out of sight when it goes to the right during the oscillatory behaviour. ACT-DRIVING then only perceives the road, which makes it slow down and go to the left again. The reader can verify in Figure 5.8 that the time annotations are further apart when ACT-DRIVING is allowed to make eye movements, than when it is not allowed to make eye movements. The higher speed also induces a higher frequency of the oscillatory behaviour. To measure this frequency we run ACT-DRIVING on a straight track for $t = 1000$ time steps, both when it is able to make gaze shifts and when it is unable to make gaze shifts. Then we measure the number of times that ACT-DRIVING makes a switch from steering to the left to steering to the right. The behaviour has an average period of 10.72 time steps when ACT-DRIVING can move its gaze and 20.96 time steps when it cannot.

One may argue that this type of gaze movement is simply *gaze stabilisation*. We prefer not to call it gaze stabilisation, for the following two reasons. First, ACT-DRIVING does not keep the line in the exact same position in its gaze window. If it *did*, then the important oscillatory behaviour of the car would not arise (the car would probably go off-track, as in Figure 5.9). Second, gaze stabilisation is typically associated with the prevention of blur (see Chapter 1). We do not model blur; the integration of optic signals in the visual feature extraction of ACT-DRIVING is assumed to be smaller than the simulation time step of 50 ms. As a consequence, ACT-DRIVING does not have to stabilise its gaze to prevent blur.

5.4.3 Avoid Disruptive Visual Inputs

The third function of the gaze shifts is to avoid disruptive visual inputs when encountering an obstacle. Figure 5.7 shows a sharp gaze shift, at $i \approx 210$ time steps. Such peaks occur when ACT-DRIVING gazes at an obstacle: it has an *eye reflex* to look away from an obstacle. To understand whether and how this eye reflex is useful, we first explain the general obstacle-avoidance behaviour of ACT-DRIVING. Then we determine whether the eye reflex contributes to ACT-DRIVING's performance. If it does, we delve into the reasons why the eye reflex improves performance.

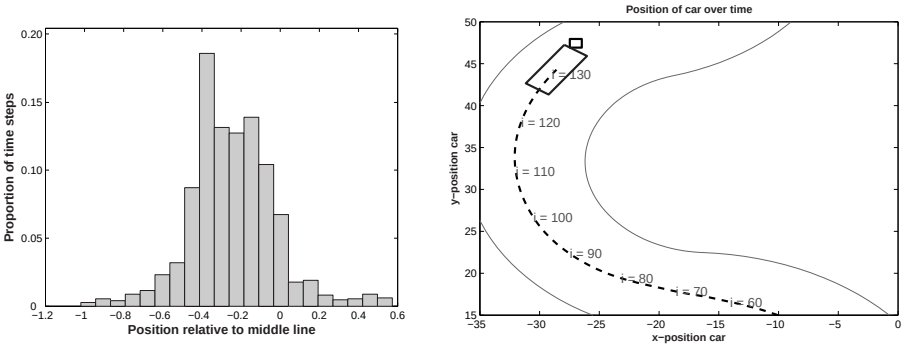


Figure 5.13: **Left:** Car position relative to the centre of the road. The histogram shows the proportion of time steps spent at different distances from the centre. Negative values represent the left part of the road, positive values the right part. **Right:** Example of a car crash, where the position of the car centre is -0.79.

Obstacle Avoidance

In general, ACT-DRIVING drives mostly on the left side of the road, and only actively avoids obstacles on that side. The left side of Figure 5.13 shows the position of the car relative to the middle line over time, on a curved track. On such a track it spends most of its time to the left of the middle line[8]. This is in contrast to a straight track, on which it drives almost exactly on the middle line (slightly right of it). By driving in the middle, the car can actually avoid the obstacles on *both* sides with small margins, but this strategy does not seem possible for curved tracks. Therefore, ACT-DRIVING chooses to drive on the left side and actively avoids obstacles on that side. The obstacles on the right side are then automatically avoided as well. The right side of Figure 5.13 shows ACT-DRIVING crashing into an obstacle. The position of the car centre at the moment of the crash is -0.79.

Contribution to Performance

Does the eye reflex help ACT-DRIVING to improve its performance on the driving task? One could doubt about this, since the visual inputs to ACT-DRIVING are much higher than usual when it is looking at an obstacle. The eye reflex may be a spurious side-effect of evolution. To test this, we manipulate ACT-DRIVING's gaze shifts in a subtle manner. We disable the gaze shifts only, if ACT-DRIVING is close to an obstacle on the left side *and* attempts to move the x-coordinate of the gaze further than 240 pixels.

The performance of the manipulated ACT-DRIVING is shown in the fourth row in Table 5.2 ('No eye reflex'). The average performance is 0.162, which is significantly lower than in the normal condition. This implies that the eye reflex is useful.

[8]We note that there is no rule dictating ACT-DRIVING to drive on the left. The driving behaviour is a result of evolution, not a sign that ACT-DRIVING is British. In Subsection 5.4.1 we mentioned that this is probably due to the tachometer obstructing the view on the right.

Below, we investigate three possible purposes of the gaze shifts during the eye reflex: (1) to gather information, (2) to influence the proprioception of ACT-DRIVING, and (3) to prevent overreactions of the car controller.

(1) Information Gathering

An obvious explanation of why the eye reflex is useful, is that ACT-DRIVING 'looks' where it is going. If it encounters an obstacle, it shifts its gaze to the right in order to verify whether the road is free, where the road is going, etc. Although this explanation seems plausible at first sight, it does not make much sense in the context of our experimental setup. For example, the road is always free of obstacles just after another obstacle, since we ensure a certain margin between subsequent obstacles. Therefore looking for an obstacle has no evolutionary advantage for ACT-DRIVING. In addition, the gaze locations during the eye reflex do not seem to be informative on where the road is going. Many of the gaze locations are on the centre of the road, locations at which only subtle gray levels may indicate the direction. So it is unlikely that ACT-DRIVING shifts its gaze to discover the direction of the road.

We want to know whether ACT-DRIVING makes the gaze shifts during the eye reflex to gather information from the environment, or whether it makes these shifts for another reason. Of course, if the visual inputs are always exactly the same during the eye reflex, then there would be no information in these inputs. Stated differently, if there is no bottom-up information in the inputs, there can be no task-relevant information either. However, the visual inputs during the eye reflex show small differences. Evaluating ourselves whether these small differences are informative raises a subjectivity problem: the visual inputs may not be informative to us, while they are informative to ACT-DRIVING.

To establish whether the small input differences convey information to ACT-DRIVING on the current state of the road, we use the third type of manipulation (see Subsection 5.2.3)[9]. First, ACT-DRIVING performs $n = 1000$ runs on various curved tracks under normal conditions. During these runs, we store the inputs of ACT-DRIVING when it makes an eye reflex, annotating them by the corresponding gaze location. Subsequently, ACT-DRIVING performs $n = 1000$ new runs. But this time, when it makes an eye reflex, we do not extract features from the simulator view. Instead, at each time step during the reflex we provide it with a visual input that was gathered during the first 1000 runs. We determine the $k = 10$ stored inputs that are closest to the current gaze location and select one of these inputs at random. As a consequence, ACT-DRIVING receives visual inputs that are plausible for the current gaze location, but that cannot convey any information on the current state of the road. As mentioned before, if ACT-DRIVING actually extracts information from the visual inputs, its performance should degrade when replacing the current inputs with the stored inputs.

The performance of ACT-DRIVING when we replace the visual inputs during the eye reflex is shown in the fifth row in Table 5.2 ('Replace inputs'). It is 0.173, which is only 0.01 lower than the normal performance. This difference is not statistically significant. In other words, the gaze shifts do not serve to gather information.

[9] We note that the manipulation evades the subjectivity problem mentioned before.

(2) Influence Proprioception

Our experimental manipulation established that ACT-DRIVING does not gather information from the environment during the eye reflex. This finding leads us to investigate another possible reason for the eye reflex. When ACT-DRIVING looks to the right, its proprioceptive inputs change. It may use these inputs to achieve successful behaviour. In order to investigate whether the change in proprioceptive values is the reason for the success of the eye reflex, we fixated the proprioceptive inputs during the eye reflex. The resulting average performance is 0.173 (see the sixth row in Table 5.2, 'No proprioc.'). The difference with the normal condition is not statistically significant. Therefore, the change in the proprioceptive inputs cannot explain the success of the eye reflex.

(3) Prevent Overreaction Car Controller

We argue that the reason for the eye reflex cannot be found in what happens under normal conditions: it can be found in what does *not* happen. When ACT-DRIVING cannot make the eye reflex, it goes off-track more often after an obstacle. ACT-DRIVING has a strong steering reaction to the obstacle. If it does not look away from an obstacle, it goes more to the right when deviating from the normal driving path. As a consequence, ACT-DRIVING goes faster when it goes to the left after the obstacle. Sometimes it goes too fast to the left and ACT-DRIVING does not succeed in re-entering its oscillatory behaviour. The gaze window overshoots the road line and ends up in the grass. This usually results in ACT-DRIVING going off-track. The difference in the car-trajectory with and without eye reflex can be seen in Figure 5.14. The square represents the obstacle. The dashed line is the trajectory of ACT-DRIVING with eye reflex. The solid line is the trajectory of ACT-DRIVING without eye reflex. ACT-DRIVING deviates more from its path when it does not have the eye reflex. Although the deviation does not seem excessively large, it often creates problems when ACT-DRIVING attempts to relocate the road line.

The observation above leads to the explanation that ACT-DRIVING avoids gazing at obstacles, since the resulting visual inputs disrupt its driving behaviour.

5.5 Discussion

In Subsection 5.4.3 we showed that ACT-DRIVING does not always make gaze shifts in order to gather information from the environment. Below, we first discuss two possible criticisms regarding this finding. Then, we indicate its possible theoretical and practical implications.

Criticisms

Here, we discuss two possible criticisms on the current study: (1) the eye reflex is not crucial to the behaviour, and (2) we did not provide any evidence for the type of movement being generalised to other active vision models or tasks. We address these two criticisms in turn.

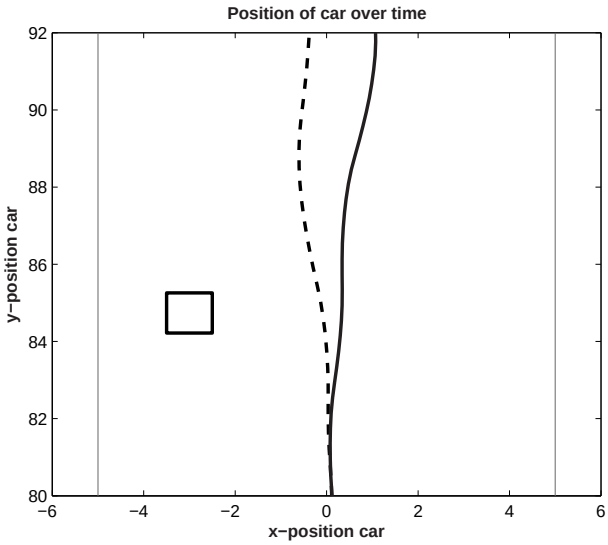


Figure 5.14: Car-position near an obstacle on a straight track. The square represents the obstacle. The dashed line is the trajectory of ACT-DRIVING with eye reflex. The solid line is the trajectory of ACT-DRIVING without eye reflex. The road goes from $x = -5$ to $x = 5$.

We do not claim that the eye reflex is as crucial to ACT-DRIVING's behaviour as finding the line or keeping the line in sight. However, we do think that the eye reflex is a movement worth analysing, for the following three reasons. First, it is the most salient type of eye movement that ACT-DRIVING performs. Second, the performance improvement of the eye reflex may be smaller than the other types of movements, but it does improve the performance significantly[10]. Third, the eye reflex may play only a subtle role in ACT-DRIVING, but similar types of movements may play larger roles in other systems.

The last reason brings us to the criticism regarding the generality of the finding. We agree that we need more experiments to establish whether eye movements in other systems also serve to prevent or provoke certain behaviours. However, we believe that these experiments will support the current finding. Namely, adaptive active vision models are known to take actions that reduce the complexity of internal processing (see, for instance, Chapter 4, or Nolfi and Marocco, 2002). It is therefore to be expected that such adaptive models rather avoid visual stimuli that are difficult to process, than evolve suitable internal processing to handle them. In the concrete terms of our experiment: it may be harder to set the neural network parameters in order to restrict the reaction to completely white inputs, than it is to avoid such inputs altogether.

[10] In this sense it may be compared with the choice between all-season tires and winter tires. All-season tires have less grip on the road, when it is cold and slippery. Although one does not crash immediately when employing all-season tires in such conditions, it is preferable to use winter tires.

Possible Implications

If the finding does turn out to be more general, then it can have both theoretical and practical implications. The finding is theoretically interesting, since it refutes the assumption that gaze shifts only serve to gather information. This assumption is central both to computer vision models and to models of human and animal vision. Let us sketch a possible implication of our finding by providing an example of human vision. Wilkie and Wann (2003) and Wann and Wilkie (2004) have extensively investigated human gaze behaviour during driving. They have investigated how the driving behaviour of the subjects was influenced by the retinal optic flow, the visual angle between the car body and a target location, and extraretinal signals. Wann and Wilkie (2004) discussed that driving manuals instruct humans to look away from approaching cars, and instead look at the road next to these cars. It is better to look away from an approaching car, because we have a tendency to go where we are looking. Wann and Wilkie (2004) explained that we have this tendency, since it gives us most information in the optic flow on where we are going. Our finding opens up the possibility that we have this tendency for other reasons: the only goal of the gaze shifts may be to suppress an unwanted tendency.

The finding is practically interesting, since it may imply that we can improve an active vision model if we adapt the gaze shifts to the rest of the model's behaviour. In our study, it helps to adapt the car controller and the eye controller together, since it allows them to adapt to each other's capabilities and shortcomings. It is not common to look at a control task involving gaze shifts in this way. For example, in a probabilistic approach to obstacle avoidance the gaze shifts serve to gather information on the state of the world. The state estimate is then used to take other, non-visual actions. Training these two parts of the model independently may result in suboptimal performance.

5.6 Chapter Conclusion

In this chapter, we set out to answer RQ 3: *How does an adaptive gaze control model use its gaze shifts in a control task?* To answer the question, we studied the gaze shifts of the best evolved instance of ACT-DRIVING. From our analysis we may conclude that its gaze shifts have the following three functions: (1) to find relevant visual features in the beginning of a run, (2) to keep relevant visual features in sight, and (3) to avoid disruptive visual inputs. Where the first two functions concern the gathering of information from the environment, the third function does not. This corroborates our conjecture at the beginning of the chapter: the gaze shifts of adaptive active vision models also serve other purposes than the gathering of information from the environment.

In the current and previous chapter, we mainly focused on the analysis of adaptive active vision models. Although we adapted the models to challenging visual tasks, we did not compare the performance of the models to that of other state-of-the-art computer vision models. Therefore, the question arises whether adaptive active vision models present a viable alternative for such models. We focus on this issue in the next chapter.

Chapter

6

Gaze Control for Object Detection

This chapter is partly based on the following publications[1].

1. G.C.H.E. de Croon and E.O. Postma (2006). Active object detection. *Belgian-Dutch AI Conference (BNAIC 2006), Namur, Belgium* (eds. P.-Y. Schobbens, W. Vanhoof, and G. Schwanen), pp. 107 – 114.
2. G.C.H.E. de Croon and E.O. Postma (2007). Sensory-motor coordination in object detection. *1st IEEE Symposium on Artificial Life, Honolulu, HI* (ed. H.A. Abbass), pp. 147 – 154.
3. G.C.H.E. de Croon (2007). Active object detection. *2nd International Conference on Computer Vision Theory and Applications (VISAPP 2007), Barcelona, Spain* (eds. A. Ranchordas, H.J. Araújo, and J. Vitrià), pp. 97 – 103.

In this chapter, we shift our focus from pure analysis to the performance of an adaptive active vision model on a challenging visual task. To this end, we investigate RQ 4: *Can an adaptive gaze control model perform on a par with state-of-the-art computer vision models on the task of object detection?* To answer this question, we apply a gaze control model to two object-detection tasks in natural static images: face detection and car detection. This gaze control model is an instantiation of ACT-FRAME, and is referred to as ACT-DETECT. The performance of ACT-DETECT is compared with that of existing methods for object detection. Below, we first motivate the use of a gaze control model for object detection and then provide an overview of the remainder of the chapter.

Humans and Object Detection

Humans employ the context of an object to detect it quickly and accurately in a visual scene. Violations of the context diminish the speed and accuracy of the detection (cf. Biederman, Glass, and Stacy, 1973; Biederman, Mezzanotte, and Rabinowitz, 1982; de Graef, Christiaens, and d'Ydevalle, 1990; Henderson, Jr., and

[1]The author would like to thank the publishers and his co-author for their permission to use parts of the publications in this chapter.

Hollingworth, 1999). The reader can easily verify the truth of this statement by (1) attempting to find a face in Figure 6.1, and then (2) attempting to find a face in Figure 6.2. To detect an object, humans exploit its (statistical) relations to other scene elements (Biederman *et al.*, 1973; Palmer, 1975; Biederman *et al.*, 1982; de Graef *et al.*, 1990; Chung and Jiang, 1998; Henderson *et al.*, 1999; Oliva *et al.*, 2003; Neider and Zelinski, 2006). The relations between scene elements are due to the structure of the world. In Figure 6.1 we have some difficulty in finding the face, because the usual structure is absent. Both coarse-scale regularities (e.g., the relation of the side walk to possible face positions) and fine-scale regularities (e.g., the relation of feet to possible face positions) have been perturbed. In the original scene depicted in Figure 6.2 we can use both coarse-scale and fine-scale scene elements to gain information on possible object locations. By using these elements we can locate objects without scanning the entire image, as we end up doing in Figure 6.1.

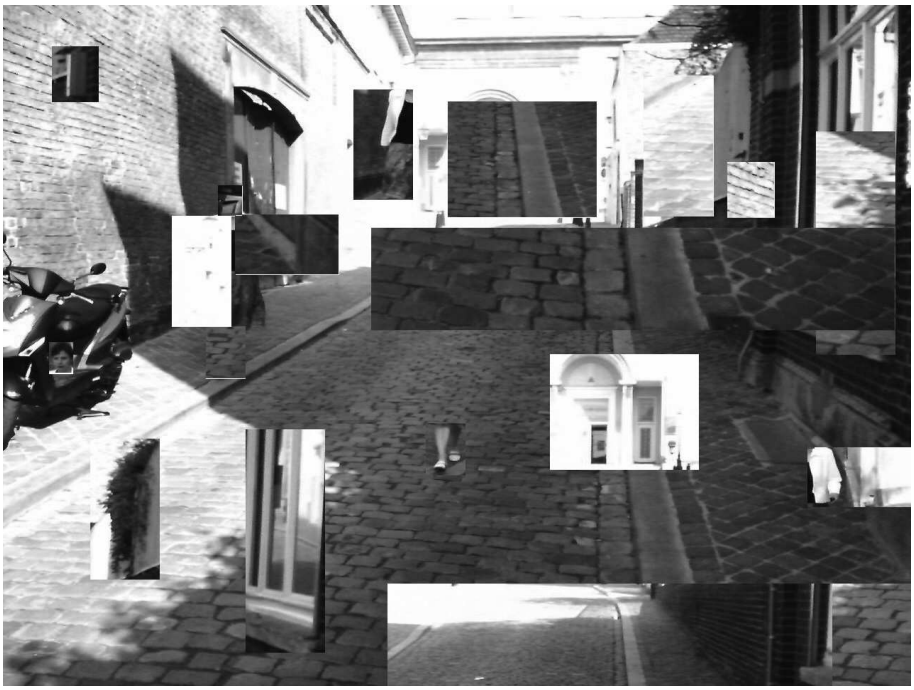


Figure 6.1: Find the face in the image (1).

Passive Scanning

Most existing artificial object-detection methods focus on the detection of features of the object itself (cf. Rao *et al.*, 1995; Moghaddam and Pentland, 1997; Burl, Weber, and Perona, 1998; Rowley, Baluja, and Kanade, 1998; Papageorgiou and Poggio, 2000; Schiele and Crowley, 2000; Viola and Jones, 2001). Such methods are general in the sense that they neither make assumptions on the image content outside of

the object area, nor on the prior distribution of the object locations. They *passively* scan the entire image: local image samples extracted during scanning are not used to guide the scanning process. As a consequence, the scanning process is inefficient. We mention two main types of object-detection methods that employ passive scanning: window-sliding methods and constellation-based methods.

Window-sliding methods (e.g., Osuna, Freund, and Girosi, 1997; Rowley *et al.*, 1998; Sung and Poggio, 1998; Schneiderman and Kanade, 2000; Viola and Jones, 2001; Kruppa, Castrillon-Santana, and Schiele, 2003; Li and Zhang, 2004; Torralba, Murphy, and Freeman, 2004a) employ passive scanning to check for object presence at all locations of an evenly spaced grid. They extract a local sample at each grid point and classify it either as an object or as a part of the background.

Constellation-based methods (e.g., Burl *et al.*, 1998; Weber, 2000; Weber, Welling, and Perona, 2000a; Weber, Welling, and Perona, 2000b; Fergus, Perona, and Zisserman, 2003; Dorkóand Schmid, 2003; Loeff *et al.*, 2005; Holub, Welling, and Perona, 2005; Mikolajczyk, Leibe, and Schiele, 2006; Carbonetto *et al.*, in press - online; Fergus, Perona, and Zisserman, 2007) detect objects by detecting constellations of their parts. They generally proceed in two stages: interest-point detection and constellation evaluation. For the first stage they employ passive scanning to determine *interest points* in an image. They calculate an interest value for local samples at all points of an evenly spaced grid. Examples include the calculation of the entropy of gray values at multiple scales (Kadir and Brady, 2001), and difference-of-Gaussian responses (in the SIFT-approach by Lowe, 2004). Subsequently, at the interest points, the methods extract new local samples that are either identified as a part of the object or as a part of the background. An object is recognised if there is a constellation of recognised parts that is sufficiently similar to a learned constellation object model[2].

There is a growing number of object-detection methods that uses both the features of the object itself and the features of its context. The use of context has as goal to improve the detection performance in terms of the robustness, the computational efficiency, or both. These improvements come at a price of a lower application generality, because the detections become dependent on the context. We distinguish two existing types of such methods.

The first type uses the appearance of the image region surrounding the object (Kruppa *et al.*, 2003; Bergboer, Postma, and van den Herik, 2004; Hoiem, Efros, and Hebert, 2005; Hoiem, Efros, and Hebert, 2006; Wolf and Bileschi, 2006; Perko and Leonardis, 2007; Elder *et al.*, 2007) or the presence of nearby objects and scene elements (Bar, 2004; Torralba, Murphy, and Freeman, 2004b; Wolf and Bileschi, 2006). This type of method generally improves the robustness of detections. The object-detection method in Kruppa *et al.* (2003) does not only use the appearance of the face, but also the appearance of the head and shoulders. This use of context reduces the number of *false positives*, i.e., the number of times that the method classifies a part of the background as a face. In both Kruppa *et al.* (2003) and Bergboer *et al.* (2004), the appearance of the context is also used to speed up detection. The method first locates plausible object contexts on a coarse scale, and then verifies object pres-

[2]We note that there are also methods that apply a detector of an entire object at the interest points, e.g., Mitri *et al.* (2005).

ence on a finer scale.

The second type of method uses the global image appearance to facilitate object detection. It uses a holistic representation of the image in order to determine the probability distribution over the possible object locations[3] (Torralba, 2003; Murphy *et al.*, 2006; Torralba *et al.*, 2006). In Murphy *et al.* (2006) the global appearance is used to determine a region of interest in the scene, in which objects are expected to occur. Within this region, passive scanning is applied to perform the final object detection.

We note that all of the above-mentioned contextual methods employ passive scanning at some stage of the detection process: they do not use the information contained in local samples to guide their scanning process.

Figure 6.2: Find the face in the image (2).

Active Scanning

We aim at more efficient object detection by employing an adaptive gaze control model that *actively* scans the image: it uses local image samples to make gaze shifts that should go towards likely object locations. In the active scanning process, it should avoid regions unlikely to contain the object. At the final scanning location, the model verifies object presence with a classifier. The goal of active scanning is

[3]The representation can also be used to determine the probability of object presence and the probability distribution over different object sizes.

to save computational effort, while retaining as good a detection performance as possible.

A lower computational effort is an advantage for object-detection tasks subject to time constraints. We provide two examples of such tasks. The first example is object detection on a miniature robotic platform (cf. Roberts *et al.*, 2007). Miniature robots have rather limited processing capabilities and energy, but need to react in real time to their environment. The second example is object search in large image databases. A computationally efficient object-detection method may make it possible to search objects in the images of the database and return found objects within acceptable time limits.

Our object-detection method ACT-DETECT (see Section 6.3) is inspired by previous studies in active vision. Ballard (1991) suggested to use active scanning for object detection. He discussed the manner in which a gaze control model could make use of contextual cues (such as a table in view) to facilitate the detection of an object (such as a mug). Rao *et al.* (1995), Rao *et al.* (1997), Young *et al.* (1998), Smeraldi *et al.* (1999), and Minut and Mahadevan (2001) introduced methods to scan a visual scene actively for detecting a specific object. These methods shift their gaze to the position in their field of view that is most similar to the object. Schmidhuber (1990), Kato and Floreano (2001), Marocco and Floreano (2002), and Floreano *et al.* (2004) investigated methods that scan an artificial image actively to find black geometrical shapes (triangles or squares) on a white background, under various amounts of noise. In more recent work, Vogel and Murphy (2007) presented a reinforcement learning method that learns how to use large objects to detect smaller ones. Their experiments involve simulated object detectors, which return probabilities of detections on the basis of the ground truth labels in the images. The experimental results show that a reinforcement learning method which plans its actions for multiple time steps outperforms methods with greedy action strategies. However, the look-ahead scheme that they use is computationally expensive.

Our work is different from the studies mentioned above mainly in five ways. First, ACT-DETECT focuses on searching for objects of an object class, while most previous methods focus on searching for a specific object. Second, ACT-DETECT exploits the context of an object, instead of only exploiting the appearance of the object itself. Third, ACT-DETECT learns the relevance of different contextual cues on the basis of the image data, and is not predefined to look for specific large objects. Fourth, we apply ACT-DETECT to challenging object-detection tasks involving natural static images. Fifth, we compare the performance of ACT-DETECT with that of standard object-detection methods in computer vision.

Chapter Outline

We investigate four questions that are of importance to the success of ACT-DETECT. These questions are sub-questions of RQ 4.

RQ 4a: *Do local image samples contain information on the locations of objects in the image?*

RQ 4b: *How does the detection performance of* ACT-DETECT *compare to that of passive*

scanning methods?

RQ 4c: *How does* ACT-DETECT *select sensible gaze shifts?*

RQ 4d: *How computationally efficient is* ACT-DETECT *compared to passive scanning methods?*

Our investigation starts with RQ 4a in order to verify whether active scanning is possible. We attempt to answer RQ 4a by performing the *information experiment* (Section 6.1). In this experiment, human subjects are instructed to find objects in images through active scanning. After ascertaining that humans can exploit the information contained in local image samples to search efficiently for objects (Section 6.2), we turn to an artificial system of active scanning. Our active scanning method ACT-DETECT is introduced in Section 6.3. An evolutionary algorithm adapts ACT-DETECT to a face-detection task and to a car-detection task. These tasks allow us to answer RQ 4b, 4c, and 4d. In Section 6.4, we explain the face-detection task. Subsequently, in Section 6.5, ACT-DETECT's performance is compared with that of standard window-sliding methods (RQ 4b). Then, we analyse the solution found by the evolutionary algorithm in Section 6.6 (RQ 4c). In the subsequent sections, we do the same for the car-detection task. The task is explained in Section 6.7, the performance of ACT-DETECT is compared with that of a window-sliding method in Section 6.8 (RQ 4b), and the solution is analysed in Section 6.9 (RQ 4c). Then, we compare the computational effort of ACT-DETECT with that of window-sliding methods in Section 6.10 (RQ 4d). We discuss our results in Section 6.11 and draw our chapter conclusion in Section 6.12.

6.1 Setup Information Experiment

In the *information experiment*, we study RQ 4a: *Do local image samples contain information on the locations of objects in the image?* To answer RQ 4a, we conduct an experiment with human subjects who have to find objects by actively scanning the image on the basis of local image samples. In this section, we explain the experimental setup for the information experiment. The next section contains the report on the experiment and the analysis of the experimental results (Section 6.2).

In the information experiment, human subjects have to find cars in a subset of images of the publicly available CBCL StreetScenes database[4]. The images in the database are in colour, and we do not transform them to gray-scale. Since this thesis is not printed in colour, the figures do not represent the experimental setting accurately. The database contains images of different street scenes, with cars of various sizes and viewpoints. The experiment involves 12 human subjects, all with normal or corrected to normal vision. Four of the subjects are female, eight of them are male. All of them are Ph.D. students and in the age range of 23 to 30 years. For each subject, the experiment consists of three phases: (1) looking at example images, (2)

[4]The set and annotations can be downloaded at http://cbcl.mit.edu/software-datasets/streetscenes/. The experimental scripts (written in MATLAB©) can be downloaded at http://www.cs.unimaas.nl/g.decroon/download.php.

practicing the detection task on five different training images, (3) performing the detection task on twenty different test images.

In phase 1, the subject is shown five example images from the data set in order to obtain an impression of the type of images that he will encounter during the experiment.

In phase 2, the subject performs five training *runs* on different images. At the start of each run, the subject is provided with a small square patch from the image. This patch has a size of 128×128 pixels, and is taken at a random location in the 1280×960 image. Since we provide the pseudo-random number generator with a fixed seed, all subjects encounter the same starting locations for the same images. This allows us to compare the results between the different subjects. The left part of Figure 6.3 shows the window presented to the subject. It consists of an image patch in the centre that is surrounded by arrows. The subjects can shift their gaze over the image by clicking on the arrows. Each arrow represents a small shift in the image, of the size of the shown image patch. Hence, two arrows to the left result in a large gaze shift to the left (twice the size of the image patch shown in the centre). The goal is to find a car in the image in as few gaze shifts as possible. If a subject recognises a part of a car in the image patch in the centre, he has to click on the centre patch. If he has the idea that the image does not contain any car, he also has to click on the centre patch, no matter what it looks like[5]. When a subject tries to perform a gaze shift beyond the border of the image, the centre patch and the scanning location do not change. The middle part of Figure 6.3 shows a typical run of one of the subjects: the top row of the figure shows the centre patch and the bottom row shows what the subject clicked on. The subject recognised a car part after performing four gaze shifts. At the end of each run, the entire image is shown to the subject, with the image patches (solid boxes), gaze shifts (arrows), and labelled cars (dashed boxes). For the example run, this is illustrated in the right part of Figure 6.3. Most subjects only need one or two training runs to understand the experiment.

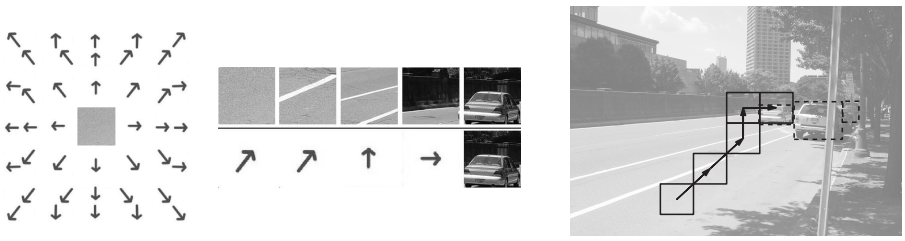


Figure 6.3: Illustration of experimental setup for the information experiment. **Left:** window shown to the subject. **Middle:** A typical run. The top row contains the image samples shown to the subject, the bottom row shows on which arrows the subject clicked. **Right:** At the end of a run, the entire image is shown to the subject.

[5]Note that if a subject clicks on a patch not containing any part of a car, we assume that he believes no object is present in the image.

In phase 3, the subjects perform twenty testing runs. Each testing run takes place in a different image. During each run, we monitor the shifts made by the subjects. This allows us to keep track of the number of shifts made per subject per image. We also monitor whether the subjects correctly detect a car or correctly indicate the absence of cars. Figure 6.4 shows all twenty test images.

Figure 6.4: The twenty test images for the information experiment.

6.2 Results Information Experiment

All subjects were able to detect the cars in the testing images with a small number of image samples (≤ 5). Figure 6.5 shows a histogram of the number of samples used by the subjects per image. The histogram is based on all runs of the subjects on the entire testing set. The figure clearly shows that in most images, fewer than five samples are sufficient to locate a car. This corresponds to less than 7% of the entire image (each local image sample represents $\approx 1.33\%$ of the image's surface). Most of the runs with more than ten samples are associated with the two images in the test set that contain no car (image 4 and 13, see Figure 6.4). In these cases, most subjects start an exhaustive search of the road. They abandon the search after a certain number of samples, by clicking on the centre patch.

The recorded gaze trajectories suggest that the subjects are exploiting the information in the local samples to find short paths to likely car locations (see, for

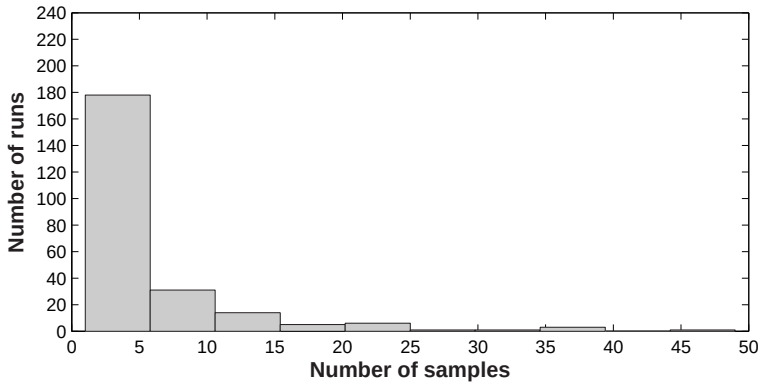


Figure 6.5: Histogram of the number of samples for all 12 subjects' runs on the entire set of testing images.

example, Figure 6.3). Therefore, we expect the success of the human subjects to be caused by an intelligent gaze shift strategy.

However, there is a possibility that the success of the subjects may be more due to their capacity of recognising car parts than their capacity of selecting intelligent gaze shifts. To investigate this, we employed a model that perfectly recognises car parts, but that determines gaze shifts in an unintelligent manner. Specifically, we used a *random gaze shift model*. The model selects random gaze shifts, but is provided with a car-part-detecting *oracle*. This oracle ends the run as soon as the model is gazing at a car part, basing itself on the ground-truth labelling. If the model is less successful than the human subjects, then the success of the human subjects cannot be solely explained by the successful recognition of car parts. It is then shown that the human subjects exploit the information in local image samples to shift towards probable car locations.

For a fair comparison with the human subjects, we do not apply the random gaze shift model to images without cars (4 and 13). In addition, for a fair comparison of the variance between different runs with the human subjects, we apply the random model 12 times to the testing set. The model encounters the same images and associated starting locations as the human subjects did.

Figure 6.6 shows the average number of samples (and standard error of the mean) per image, for both the human subjects ($\times$) and the 12 applications of the random gaze shift model with oracle (diamond). For image 20 the average number of samples for the random model is 75 (outside of the figure's scope). From the results, we may make three observations.

First, the human subjects outperform the random model on 12 out of the 18 images with cars. For 7 out of the 12 cases the performance differences are significant. To determine significance, we used a randomisation test with $p < 0.05$ (Cohen, 1995). The difference between human and random performance depends on the image. On closer inspection we noticed that there are three factors that seem to be important for this difference: (a) the proportion of the image covered by cars, (b)

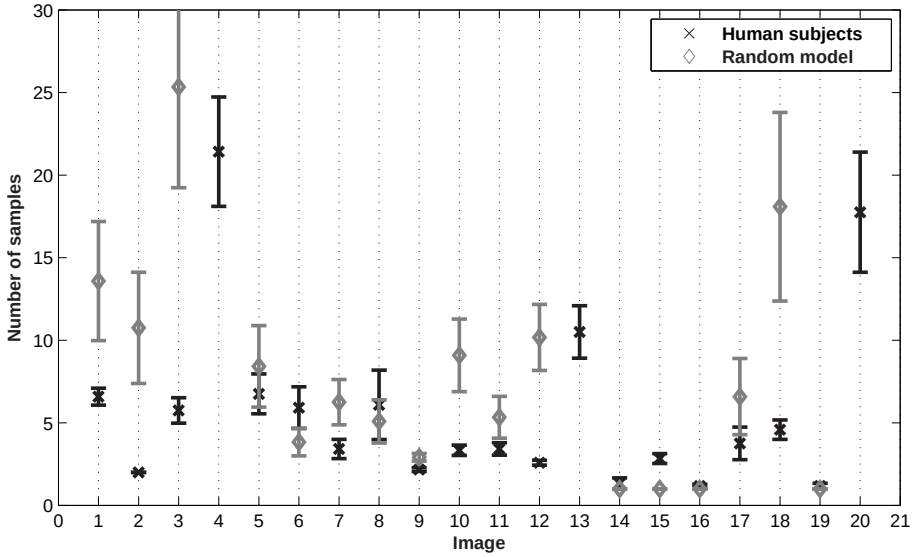


Figure 6.6: Average number of samples and standard error of the mean per image, for the twelve human subjects ($\times$) and the twelve applications of the random gaze shift model with oracle (diamond).

the number of cars, and (c) the distance of the starting location to the closest car. The random gaze shift model performs worse than the human subjects, when there are (a) smaller and (b) fewer cars in the image, and (c) when the starting location is further from the closest car. For example, the cars in image 3 (Figure 6.4) cover only $\approx 3.1\%$ of the image, and the starting location is relatively far away from the closest car (501 pixels). The small portion of the image covered by the cars, and the large distance to the starting position, results in a bad performance of the random model. In contrast, the human performance is not affected by these difficult conditions.

Second, the standard errors of the mean are larger for the random model. The human subjects seem to exhibit little variance in the number of samples that they need for detecting a car. This suggests that their gaze shift strategies are similar.

Third, we observe that the random model has a better average performance than the human subjects on some of the images (6, 8, 14, and 15). This is caused by the model's oracle. For example, the first sample of image 15 shows the curb of a street with a horizontal line in the upper part of the sample. Most subjects shift the gaze upwards to verify whether the line is part of a car. The oracle does not need this extra gaze shift.

In summary, the results of this experiment indicate that local samples contain information on object locations in an image. In what follows, we verify whether it is possible for an artificial gaze control model to exploit such information.

6.3 ACT-DETECT

In this section, we introduce our active object-detection method, ACT-DETECT. As it is an instantiation of ACT-FRAME, it extracts information from local image samples, and uses this information to shift its gaze window towards an object location. The main difference with the other gaze control models in the thesis is that ACT-DETECT extracts its inputs at different image scales. The reason for this difference in extraction is that the information on an object's location can be found at different image scales. For example, in Figure 6.2 the information is both in coarse features such as the sidewalk and in detailed features such as the feet of the person. Since the information at different image scales has to be interpreted differently, we employ different parameter values for gaze control at each of the s image scales (the choice for s depends on the task).

Figure 6.7 illustrates a run of ACT-DETECT. The left part of Figure 6.7 shows that ACT-DETECT starts at a random location at a coarse scale (in the figure, $s = 3$) in the image pyramid. Per scale, ACT-DETECT makes a fixed number of gaze shifts t (solid arrows). After these gaze shifts it proceeds to a more detailed scale (thick lines). In this manner, it progressively refines its search for an object by making successive gaze shifts further down the image's scale space. Since ACT-DETECT only takes local samples in each scale, we do not have to construct the entire scale space for each image. The right part of Figure 6.7 illustrates for the most detailed scale how the gaze control model takes a local image sample from the gaze window (box), centred at the current gaze location (x). The model extracts visual features from this sample and passes the feature values to the model's controller, which maps the features to a relative gaze shift in the image (solid arrow) to the new gaze location (o). The final gaze location at the most detailed scale is considered a candidate object location. We use a standard object detector to verify whether this location actually contains an object. Below we explain our implementation choices for the experiments in this chapter.

6.3.1 Number of Scales

On the basis of findings in an earlier study (de Croon and Postma, 2006), we employ $s = 2$ scales in our experiments. ACT-DETECT uses different parameter values for each scale, since it needs different visual features and behaviours at each scale. To facilitate our discussion of ACT-DETECT at different scales, we refer to the gaze control model with the parameters for the coarse scale as the *coarse-scale model* and to the gaze control model with the parameters for the finer scale as the *fine-scale model*. As mentioned above, the coarse-scale model starts at a random location in the image. It has a gaze window size equal to one third of the image width. ACT-DETECT makes $t = 5$ successive gaze shifts, and then it moves to the finer scale. The fine-scale model starts at the final location of the coarse-scale model, and has a gaze window of one fourth of the image width. It also performs $t = 5$ gaze shifts. Both the coarse-scale model and the fine-scale model have a visual feature extraction module and a controller module.

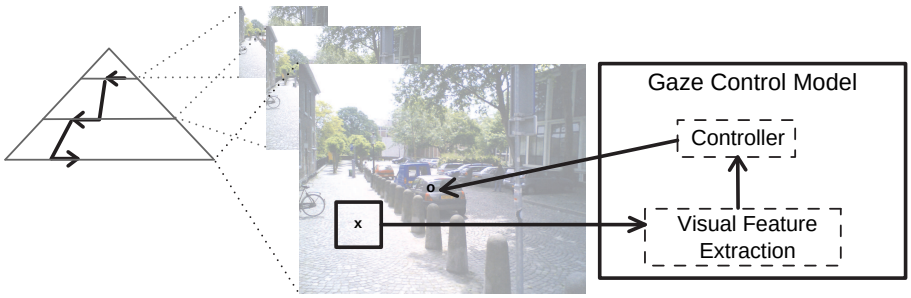


Figure 6.7: Overview of ACT-DETECT. The pyramid on the left shows that ACT-DETECT exploits visual features on s different scales by moving through the image's scale space. It performs a fixed number of gaze shifts per scale (solid arrows) and then moves down to the next scale (thick lines). In this manner, it progressively refines its search for the object. For this example $s = 3$. The right part of the figure illustrates the core concept of ACT-DETECT. It extracts visual features from a local image sample (box) at the current gaze location (x). The feature values are passed to the controller that determines a relative gaze shift in the image (solid arrow) to the next gaze location (o).

6.3.2 Visual Feature Extraction

To facilitate comparison with existing window-sliding methods, we adopt features that are often used by such methods: the *integral features* introduced in Viola and Jones (2001). The main advantage of these features is that they can be extracted with little computational effort, independent of their scale. After the preprocessing (see Viola and Jones, 2001), it is not necessary to create a scale space explicitly.

We make the application of ACT-DETECT's features more flexible than for ACT-CLASS in Chapter 4. The top row of Figure 6.8 shows the types of features used in the object-detection experiments. For each feature that serves as input to the controller, we can select both the type and the corresponding area in the gaze window. In contrast to the experiments in Chapter 4, we do not predefine the possible areas, but allow any possible area within the gaze window.

The bottom row of Figure 6.8 shows an example feature. It is of type 1 and spans a large part of the right half of the gaze window. The value of this feature is equal to the mean gray-value of all pixels in area A minus the mean gray-value of all pixels in area B. The example feature will respond to vertical contrasts in the image.

6.3.3 Controller

The controller accepts the n extracted features as input. It is a completely connected multilayer feedforward neural network, with $h = \lfloor n/2 \rfloor$ hidden neurons and $o = 2$ the output neurons. Both the hidden and output neurons have a sigmoid activation function: $a(z) = \tanh(z) = 1 - \frac{2}{1+e^{2z}}$. The two output neurons encode for the gaze shift $(\Delta x, \Delta y)$ in pixels as follows: $\Delta x = \lfloor d_{\max} \times \text{out}_1 \rfloor$, $\Delta y = \lfloor d_{\max} \times \text{out}_2 \rfloor$. The constant $d_{\max}$ represents the maximal displacement in the image in pixels.

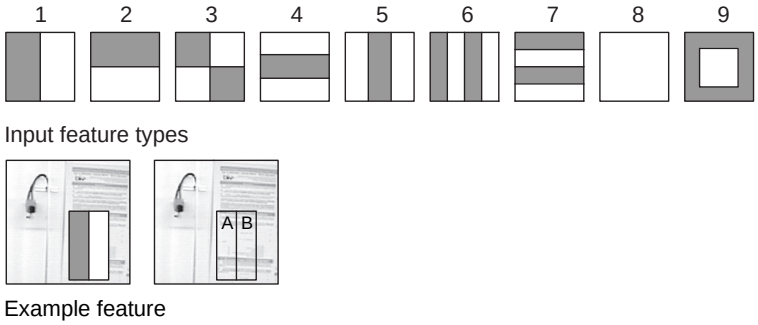


Figure 6.8: Possible feature types (top part of the figure) and an example feature shown in the gaze window (bottom part of the figure).

6.3.4 Object Detector

The verification whether the final gaze locations are located at object locations is performed by a **S**upport **V**ector **M**achine (SVM) (Osuna *et al.*, 1997; Qi, Doermann, and DeMenthon, 2001; Kim and Kittler, 2005)[6].

To train the classifier, we gathered image samples at the final gaze locations in the training set. The inputs to the classifier are also integral features (see Figure 6.8). We used an evolutionary algorithm with the same settings as will be explained in Subsection 6.3.6 to optimise the input features passed to the SVM.

We trained the SVM with the **S**equential **M**inimal **O**ptimisation algorithm (SMO-algorithm) (Platt, 1999). The algorithm required us to make two main choices, regarding the value of the training parameter c and the type of kernel. The value of c determines how well the decision boundary of the SVM fits the training set. It therefore represents a trade-off between a good fit on the training set and a good generalisation to the test set. In our experiments c was set to 10. Furthermore, we decided to use a non-linear **R**adial **B**asis **F**unction kernel (RBF-kernel) with $\gamma = 0.75$[7].

6.3.5 Adaptable Model Parameters

The coarse-scale model and the fine-scale model of ACT-DETECT have two types of adaptable model parameters: (1) the types and locations of the input features, and (2) the neural network weights. The object detector applied at the final gaze locations only has the first type of adaptable parameters.

6.3.6 Evolutionary Algorithm

We adapt ACT-DETECT to an object-detection task with a λ, μ-evolutionary algorithm (Bäck, 1996). For evolution, the image set is divided into two parts: half of

[6]Note that we only used the first stage of a Viola and Jones classifier in previous studies (de Croon, 2007; de Croon and Postma, 2007). This simple linear classifier was trained on all locations in all training images.

[7]We used a publicly available implementation of this algorithm, which can be downloaded at http://theoval.sys.uea.ac.uk/ gcc/svm/toolbox/. For a more detailed explanation of the SVM and its parameters, we refer the reader to Burges (1998) and Platt (1999).

the images is used for evolution and half of the images for testing.

We first evolve the coarse-scale model parameters, starting from uniformly distributed locations. Since one run of ACT-DETECT can only lead to the detection of one object, we always perform $R = 10$ runs per image. The coarse-scale model is evolved to optimise recall, i.e., the proportion of objects present in the training set that are detected by the ensemble of runs of ACT-DETECT. Then we evolve the parameters of the fine-scale model, which always starts from the end locations of the already evolved coarse-scale model. The fine-scale model is evolved to optimise precision, i.e., the proportion of runs that ends up at an object location. For the evolution of each model, the population size is $\lambda = 100$, and the best $\mu = 25$ genomes are selected to create a next generation. Evolution continues for $g = 300$ generations.

The genome representing the search space of the evolutionary algorithm is a vector of double values in the interval $[-1, 1]$. The first part of the genome encodes for the visual features of ACT-DETECT. Each feature is represented by five values, one for the type and four for the two coordinates inside the gaze window. The type of the feature is decoded as follows: type $= [|\text{gene}_1| \times 8] + 1$, where $[\cdot]$ is the round-function. The left coordinate of the feature in the window is decoded as: left $= \min(|\text{gene}_2|, |\text{gene}_3|) \times$ window_width. The right coordinate is then: right $= \max(|\text{gene}_2|, |\text{gene}_3|) \times$ window_width. We determine the top and bottom coordinate in the same manner, but with gene_4 and gene_5.

The second part of the genome encodes for the neural network weights. Each weight is directly represented by one value, which implies a weight range of $[-1, 1]$. The probability for a mutation in the genome is $p_{\text{mut}} = 0.04$, and for one-point crossover with another selected genome is $p_{\text{co}} = 0.5$.

After the optimisation of the fine-scale model, we gather the image samples at the final gaze locations in the training set to evolve the object detector. We use the same training parameters for the evolution of the object detector as for the gaze control models, but the genome only consists of double values encoding the visual features. The visual features of the object detector are evolved to optimise the proportion of samples correctly classified.

6.4 Setup Face-detection Experiment

The *face-detection experiment* is described in Section 6.4, 6.5, and 6.6. In the experiment, ACT-DETECT is adapted to a face-detection task. We explain the task in Subsection 6.4.1. In Section 6.5, the performance of ACT-DETECT is compared with that of window-sliding methods. Then, in Section 6.6, we analyse the evolved gaze strategy.

6.4.1 Face-detection Task

The face-detection task consists of detecting frontal faces in the publicly available FGNET image set[8]. We chose this image set, since other researchers have applied

[8]The FGNET image set is available at (http://www-prima.inrialpes.fr/FGnet/).

their methods to it and reported on the results. This allows us to compare our results with those reported in the literature, mainly obtained with variations of the (Viola and Jones, 2001) object detector.

The FGNET image set contains video sequences of a meeting room, recorded by two different cameras. For our experiments we used the joint set of images from both cameras ('Cam1' and 'Cam2') in the first scene ('ScenA'). The set consists of 794 images of 720×576 pixels, which are converted to gray-scale. Figure 6.9 shows five example images from the set. We use the ground truth data that is available online, in which only the faces with two visible eyes are labelled. For evolution, the image set is divided into two parts: half of the images is used for testing and half of the images for evolution. We perform a two-folded test to obtain our results, running one evolution per fold.

Figure 6.9: Five example images from the FGNET image set.

For the face-detection task, we provide ACT-DETECT with $n = 10$ features. Furthermore, the maximal displacement of the gaze window in pixels is $d_{\max} = 360$ for the coarse-scale model and $d_{\max} = 240$ for the fine-scale model. All the other parameter settings have been discussed in Section 6.3.

6.5 Performance Face-detection Experiment

This section focuses on RQ 4b: *How does the detection performance of* ACT-DETECT *compare to that of passive scanning methods?* To answer the question, we first show that the evolutionary algorithm finds successful scanning strategies for the face-detection task. Subsequently, the detection performance is compared with that of standard window-sliding methods. Since active scanning implies avoiding image regions in the detection process, we expect ACT-DETECT to perform slightly worse than the window-sliding methods.

The evolutionary algorithm finds successful scanning strategies for the face-detection task. Figure 6.10 shows ten independent runs of the best instance of ACT-DETECT (first fold). The arrows represent the gaze shifts of the coarse-scale model and the fine-scale model. At time step $i = 0$, all runs are initialised at random positions in the image. The coarse-scale model then makes five gaze shifts, followed by another five gaze shifts of the fine-scale model. At the end of scanning ($i = 10$) six out of ten runs have reached an object location. The local image samples at the final gaze locations are classified by the trained SVM. Circles indicate positive classifications, crosses negative classifications. The gaze shifts leading to a posi-

tive classification are black, while gaze shifts leading to a negative classification are gray. ACT-DETECT successfully detects both faces in the image.

Figure 6.10: Ten independent runs of ACT-DETECT on an image from the FGNET data set. See the text for a detailed explanation.

To assess how well ACT-DETECT performs on the FGNET image set in comparison with standard window-sliding methods, we constructed an FROC-plot[9]. It plots the recall against the average number of false positives per image. The goal of an object-detection method is to achieve a high recall and a low number of false positives.

For object-detection methods that mainly rely on a binary classifier, an FROC-plot can be constructed by varying the threshold of this classifier. Since both the gaze shifts and the object classification are of importance to the performance of ACT-DETECT, it is not evident how to construct an FROC-plot. Here we mention three factors that are of influence. First, the active scanning approach itself represents a choice for computational efficiency at the possible cost of the number of detections. The approach implies that parts of the image are avoided. Second, the fitness function is of influence on the FROC-plot. For example, the fitness function of the coarse-scale model puts an emphasis on recall, while the fitness function of the fine-scale model does so on precision. Third, the number of independent runs is positively related to the recall and false positive rate. A higher number of runs results in more detections and false positives. We use the third factor to construct the FROC-plot of ACT-DETECT, since it is the easiest factor to vary. In particular, R is set to $1, 3, 5, 10, 20,$ and 30.

Figure 6.11 is an FROC-plot of our experimental results, averaged over both folds (square markers, dotted line). In addition, the figure shows the results on the FGNET

[9] FROC stands for **F**ree-response **R**eceiver **O**perating **C**haracteristic.

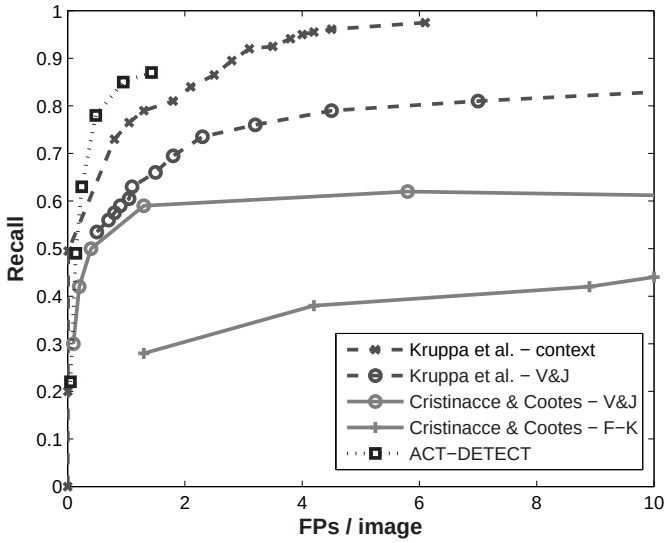


Figure 6.11: FROC-plots of the different object-detection methods for the face-detection task.

image set of the detector by Cristinacce and Cootes (2003) (solid lines), of a Fröba and Küllbeck (2002) detector (plus-markers) and a Viola and Jones (2001) detector (circular markers). It also includes the results of two Viola and Jones detectors trained on a separate image set and tested on the FGNET set (Kruppa *et al.*, 2003) (dashed lines). The first of these detectors attempts to detect face regions in the image in the same manner as the detectors in Cristinacce and Cootes (2003) (circular markers). The second of these detectors attempts to detect a face by including a region around the face, including head and shoulders (cross-markers).

Figure 6.11 leads to the observation that ACT-DETECT has a better detection performance than the window-sliding methods on the FGNET data set. This is surprising, since it avoids large image regions in the detection process. The second best method is the one by Kruppa *et al.* (2003), which takes an object's context into account. Detecting faces without considering context is difficult in the FGNET video-sequence, because the appearance of a face can change considerably from image to image (Cristinacce and Cootes, 2003). However, the context of a face (such as head and shoulders) is rather fixed. This is why approaches that exploit this context (Kruppa *et al.*, 2003; Bergboer *et al.*, 2004) have a more robust performance. The active object-detection method exploits context even to a greater extent than the methods studied in Bergboer *et al.* (2004) and Kruppa *et al.* (2003). However, ACT-DETECT does not necessarily achieve a 100% recall, if we augment the number of independent runs R or lower the threshold of the classifier.

It is interesting to note the difference between the Viola and Jones classifiers used in Cristinacce and Cootes (2003) and in Kruppa *et al.* (2003). The difference can be explained by at least three factors: a different training set, different settings

of the training parameters for the Viola and Jones classifier, and different ground truth data. In contrast to the ground truth data available online, Kruppa *et al.* (2003) also labelled profile faces.

Disregarding small differences between the experiments, the results show that ACT-DETECT performs at least as good as the window-sliding methods on the FGNET face-detection task. In the next section, we analyse how ACT-DETECT achieves this performance.

6.6 Analysis Face-detection Experiment

In this section we investigate RQ 4c: *How does* ACT-DETECT *select sensible gaze shifts?* This question is of importance, since it may help us understand the strengths and weaknesses of ACT-DETECT. To answer the question we analyse the best evolved instance of ACT-DETECT on the first fold. Our main focus in the analysis is on the coarse-scale model. We first study its evolved visual features and then its mapping from visual inputs to gaze shifts in the image. The study shows that ACT-DETECT's visual features capture contextual cues, which are exploited by its gaze shifts. Subsequently, we report on the analysis of the fine-scale model and its differences with the coarse-scale model. Then, the main findings of the analysis are summarised.

6.6.1 Visual Features

The evolved visual features capture properties of the object and its context. Figure 6.12 shows the $n = 10$ evolved input features. The features are projected on an image from the training set. They are shown at their locations and with their sizes within the gaze window (white box). The white cross indicates the centre of the gaze window. In order to interpret the evolved features, we extract them at all possible gaze locations within the image and store their values. We scale these values to obtain *feature responses* in the interval $[0, 1]$. Figure 6.13 shows a (darkened) image in the background and the feature responses on all possible gaze locations on the foreground. There are no feature responses in the border of the image, since the gaze window cannot go over the border of the image. High intensity regions mark high responses, low intensity regions low responses.

Contextual and Object Features

Figure 6.12 and 6.13 illustrate that the model uses contextual features to find the object. Let us take feature 8 as an example. This feature represents vertical contrasts to the top right of the current gaze location (Figure 6.12). We can see in Figure 6.13 that this feature has a high response when the gaze location is on the right part of a person's body. This can be used by ACT-DETECT to locate the face[10]. Besides contextual features, ACT-DETECT also exploits features that seem to be more object-related. A telling example is feature 9, a centre-surround feature located to the

[10]Other examples of contextual features are feature 4, which has a high response when ACT-DETECT is right of a person's body, and feature 7, which has a higher response on the wall than further down in the image where the persons are seated.

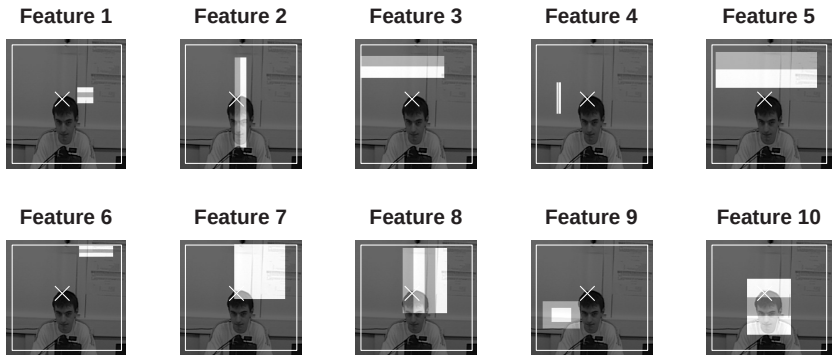


Figure 6.12: The ten evolved features for the face-detection task, shown within the gaze window (white box) of the coarse-scale model. The white cross indicates the centre of the gaze window.

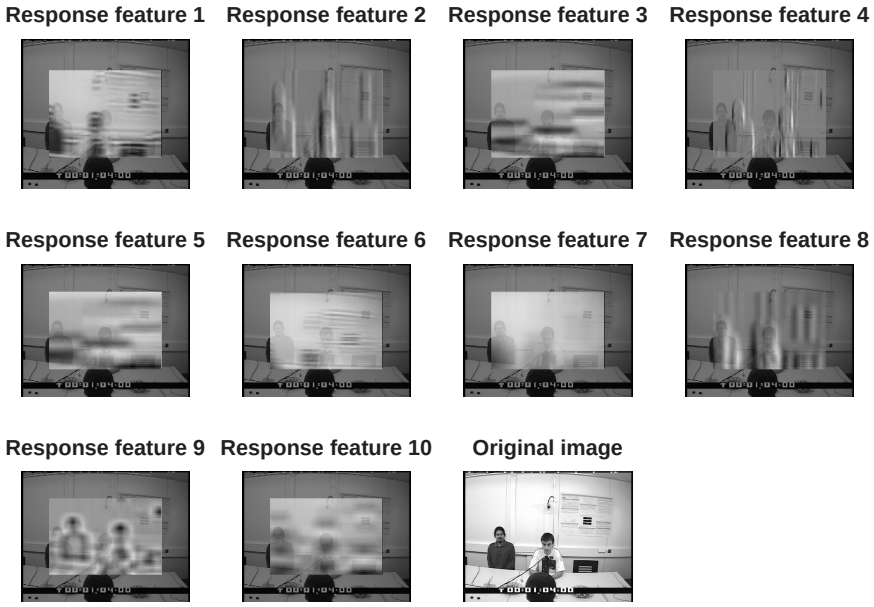


Figure 6.13: Responses of the ten features shown in Figure 6.12 in different parts of the image. The evolved features have been extracted at all possible gaze locations in an image (labelled 'original image'). Per feature, we show a darkened version of the image as the background, and the feature responses on the foreground. The feature responses are scaled to [0,1], and represented with gray-values. High intensity represents a high response, low intensity a low response. We show the response at the gaze location, i.e., the centre of the gaze window when the feature was extracted.

bottom left of the gaze location. It has a very low response if the gaze window is situated above and to the right of a head (see Figure 6.13).

Information in the Visual Inputs

To show that the evolved visual features carry information on the position of an object, we determined the relation between clusters of visual inputs on the one hand, and the horizontal and vertical displacement to an object on the other hand. Therefore, we gathered local input samples by extracting the evolved features at 30 random gaze locations for each image in a set of 145 training images (leading to 4,350 input samples), and storing the relative distance from the gaze location to the closest object. The samples were clustered with k-means clustering, for $k = \langle 2, 3, 4, \ldots, 15 \rangle$. Then, every sample was mapped to the nearest cluster centroid and the relative distances for all the samples belonging to a cluster were stored. This allowed us to determine the mutual information I between the clusters C and the horizontal displacement ΔX and the vertical displacement ΔY to the closest target: $I(C; \Delta X)$ and $I(C; \Delta Y)$. This mutual information is expressed in bits[11]. To calculate the mutual information between these variables, we binned the values of ΔX and ΔY with 10 evenly spaced bins, ranging from their minimum value to their maximum value.

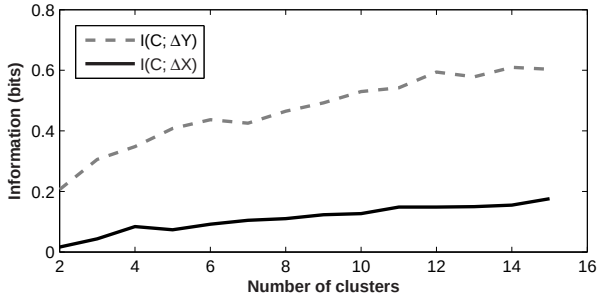


Figure 6.14: The relation between the number of clusters and the mutual information of the clustering with the horizontal distance (ΔX, solid line) and vertical distance (ΔY, dashed line) to the closest object. The mutual information is expressed in bits.

Figure 6.14 shows the calculated mutual information for different numbers of clusters. The figure leads to two observations. First, the inputs seem to have more information on the vertical distance than on the horizontal distance to the closest target ($I(C; \Delta Y) > I(C; \Delta X)$). It is probable that the environment is more structured vertically than horizontally because of gravity. Second, the information of the clustering on ΔX and ΔY grows with an increasing number of clusters. There are two main reasons why ACT-DETECT will not select the largest possible number of clusters, i.e., storing each sample with its displacement. First, an increasing refinement of the input space decreases generalisation. Second, the controller may not be capable of interpreting the input space at a very fine level. The limited neural net-

[11]Note that $I(A; B) = H(A) - H(A \mid B)$, where H is the Shannon entropy (Shannon, 1948).

work controller we use for the experiments is surely not able to store and retrieve all separate samples.

Spatial Distribution Relative to Closest Object

We can visualise the relative spatial distribution of each cluster's inputs with respect to the closest object. Figure 6.15 shows these spatial distributions for $k = 6$ clusters (the order of the clusters is irrelevant). We represent the closest object with a white cross in the centre of every inset, and high intensity regions indicate regions where the input cluster has a high probability of occurring (low intensity regions indicate the opposite). The figure shows that the clusters of inputs have different relative spatial distributions.

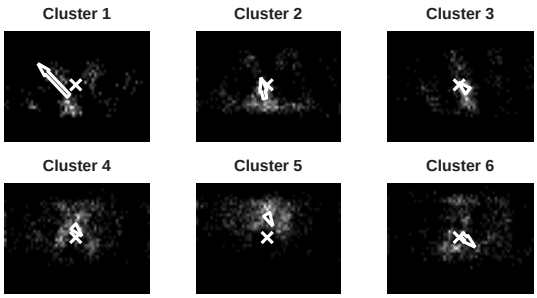


Figure 6.15: Spatial distribution of six visual input clusters relative to the closest object. High occurrence is represented with high intensity, low occurrence with low intensity. The location of the closest object is shown with a white cross. The arrows originate at the median of all locations in the cluster, and represent the actions taken by ACT-DETECT if it receives the cluster centroid as visual input.

6.6.2 Gaze Shifts

Knowing that the visual inputs contain information on the displacements to the object locations, we now turn to ACT-DETECT's mapping from its visual inputs to its gaze shifts. The analysis of this mapping will help us answer RQ 4c. Below, we first show that ACT-DETECT reacts to the visual input's relation to the closest object. Then we measure the influence of the individual features on ACT-DETECT's gaze shifts. Finally, it is analysed to which extent the actions of ACT-DETECT are determined by the prior distribution of object locations.

Shift to Closest Object

Figure 6.15 shows that ACT-DETECT exploits the relative spatial distributions of the visual inputs, while taking into account the entire chain of inputs and gaze shifts. The arrows in the figure represent the direction and size of the gaze shifts taken by ACT-DETECT when it is provided with the cluster centroids as inputs. The arrows

originate from the median relative position of the corresponding cluster of sensory inputs. Clearly, the actions depend on the relative spatial distributions of the sensory input clusters. However, ACT-DETECT does not seem to apply a greedy action policy: the ideal greedy action would shift the gaze from any position directly to the object position. For some clusters, the gaze shifts deviate from the ideal greedy action. The most remarkable deviation is that of cluster 1 for which ACT-DETECT makes a large gaze shift to the top left. This shift seems to make sense in the context of the rest of ACT-DETECT's behaviour: for the clusters above the object locations it always goes slightly more to the right. The left part of Figure 6.16 clarifies how this helps the coarse-scale model to find faces in an image. The arrows in the figure represent ACT-DETECT's gaze shifts at a 10×10 grid in the image. Generally, ACT-DETECT makes large movements to the top left for visual inputs that occur below the faces and small movements to the bottom right for those that occur above the faces. This behaviour attempts to bring ACT-DETECT close to a face, where it gets caught into a local attractor.

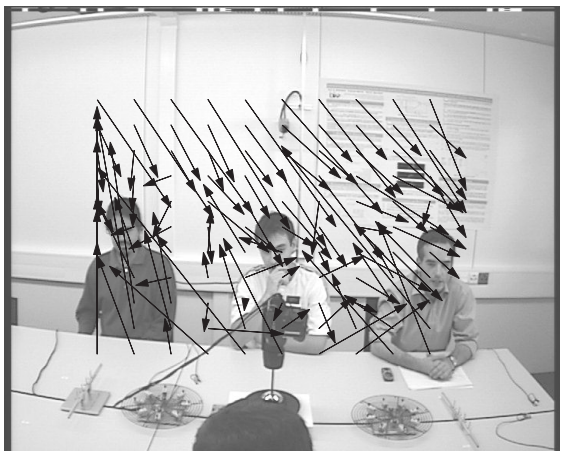


Figure 6.16: Actions taken by the coarse-scale model on a 10×10 grid in the image.

Influence of Individual Features

To determine its gaze shifts, ACT-DETECT takes into account the amount of information contained in the individual features. We can show this by determining the mutual information I between the values of each feature (F_i) and the horizontal and vertical displacement to the closest object (ΔX and ΔY). This information can then be compared with the mutual information between the feature's values and ACT-DETECT's actions A, subdivided into the horizontal and vertical part of the gaze shift, $A_{\Delta x}$ and $A_{\Delta y}$. We calculated $I(F_i; \Delta X)$, $I(F_i; \Delta Y)$, $I(F_i; A_{\Delta x})$, and $I(F_i; A_{\Delta y})$, where i is the number of the feature. To calculate the mutual informa-

tion for a feature i, a joint probability distribution was made of the feature's values F_i and the relevant second variable. We binned the variables with 10 evenly spaced bins, ranging from their minimum to their maximum value.

Table 6.1 shows the calculated information for all features. Between the features there are differences in the amount of information and in whether they give information on horizontal or vertical displacements. The table shows that in general features with more information on the displacement also have more information on the gaze shift selected by ACT-DETECT. We note that the amount of information in the features depends on the uniform sampling. Although the start locations are uniformly distributed, the evolutionary algorithm optimises the features for the entire gaze trajectory towards an object location. Therefore, features with a low information at uniformly sampled locations can be more informative if we sample closer to the objects. This is the case for feature 4. In addition, features that have a low information individually may have a high information when combined with other features within ACT-DETECT's controller.

Table 6.1: Mutual information between the evolved features F_i, $i \in \{1, 2, 3, ..., 10\}$ and the horizontal displacement to the closest object ΔX, the vertical displacement ΔY, the horizontal part of the gaze shift $A_{\Delta X}$, and the vertical part $A_{\Delta Y}$.

	$I(F_i; \Delta X)$	$I(F_i; \Delta Y)$	$I(F_i; A_{\Delta X})$	$I(F_i; A_{\Delta Y})$
$i = 1$	0.07	0.27	0.28	0.53
$i = 2$	0.10	0.05	0.10	0.09
$i = 3$	0.03	0.29	0.13	0.28
$i = 4$	0.05	0.02	0.03	0.04
$i = 5$	0.03	0.39	0.27	0.63
$i = 6$	0.07	0.14	0.24	0.22
$i = 7$	0.07	0.38	0.41	0.52
$i = 8$	0.14	0.11	0.24	0.23
$i = 9$	0.03	0.09	0.07	0.06
$i = 10$	0.06	0.32	0.27	0.27

Influence of Prior Object Distribution

The evidence above shows that ACT-DETECT uses the information in the visual features to determine its gaze shifts. However, one can wonder to what extent ACT-DETECT uses the prior distribution of object locations in the images. We refer to this prior distribution as the *location prior*. Some object-detection methods using context (such as Torralba, 2003; Wolf and Bileschi, 2006) make more use of the location prior than others (e.g., Perko and Leonardis, 2007). It depends on the application whether it is desirable to exploit the location prior or not.

Below, we show that ACT-DETECT in its current setup only makes modest use of the location prior and that it is possible to remove the influence of the location prior on the training of ACT-DETECT.

There are two main reasons why ACT-DETECT only makes modest use of the location prior. First, ACT-DETECT only receives extracted visual features and does

not employ any proprioception with information on its x, y-position in the visual scene. It can therefore only exploit the location prior indirectly. Second, we should make a difference between the prior distribution of object locations (the location prior), and the prior distribution of relative displacements to the closest object (the *relative prior*). ACT-DETECT is only sensitive to the relative prior. Although a data set may have a strong location prior, the uniform starting positions of the coarse-scale model often result in a weaker relative prior.

We can illustrate this with the FGNET image set. The location prior of the face-detection task is shown in the left part of Figure 6.17. Crosses indicate the labelled faces in the image set, in image coordinates. The FGNET data set has a strong location prior, i.e., the prior distribution of object locations in the data set is biased towards a small subset of all possible image locations. However, ACT-DETECT is set to uniformly distributed random positions at the start of each run. Therefore, it has a roughly equal probability of starting above or below and left or right of the object. We can measure the relative prior in ACT-DETECT's starting position by sampling uniformly in the image and storing the relative displacement to the closest object. For the FGNET data set the median horizontal and vertical displacement for uniform sampling are $(\Delta x, \Delta y) = (2.5, 39.0)$ in pixels. This implies that in the absence of information in the visual inputs, the best action is to move up and slightly to the right[12]. We note that many of the actions in Figure 6.16 point into different directions; the influence of the relative prior does not seem dominant for the action strategy of ACT-DETECT on the face-detection task.

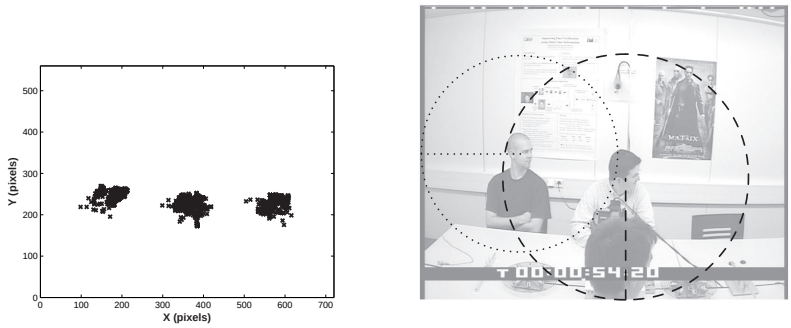


Figure 6.17: **Left:** Prior distribution of face locations in the FGNET image set. Crosses show the locations of the labelled faces in the data set in image coordinates. **Right:** Illustration of how to sample to reduce the influence of the prior. We sample inside the dotted circle for the left face, and inside the dashed circle for the right face. The lines indicate which image border is closest to the centre of the face.

Moreover, the relative prior does not prevent ACT-DETECT from detecting objects at unusual locations. Figure 6.18 shows the application of ACT-DETECT to images with unusual object locations. We use black arrows for gaze shifts that lead to a classification as object (indicated by a circle), and gray arrows for gaze shifts that

[12] Actually, the best action depends on the error measure used to define 'best'. We focus on the median, since it represents the centres of the spatial distribution clusters well (see Figure 6.15).

lead to a classification as a non-object (indicated by a cross). The left image comes from the FGNET set. ACT-DETECT finds the face of a standing person. Two out of ten runs in the image successfully localise the face, resulting in one detection. The face is not labelled, since the eyes are not visible from the camera viewpoint[13]. The right image is taken in our office in Maastricht. The image has a similar appearance to the images in the FGNET data set, due to the similarity in structure and appearance of different offices. Two out of ten runs in the image successfully localise the face, resulting in one detection. The figure shows that the relative prior does not prevent ACT-DETECT from finding a face of which the position is different from FGNET's location prior.

If desired, one can eliminate the relative prior by changing the sampling of starting locations. Such a sampling strategy is illustrated in Figure 6.17. The figure shows that each object has an associated *sampling circle* that goes through the closest border of the image. We sample within this circle to remove the relative prior. Any shape that is symmetrical around both the x-axis and y-axis when centered on $(0, 0)$ will remove the relative prior. When sampling within such a shape the median relative displacement is $(0, 0)$, since $\forall \Delta x : p(\Delta x) = p(-\Delta x)$ and $\forall \Delta y : p(\Delta y) = p(-\Delta y)$. In addition, a circle equalises the probability for all different directions. Whereas uniform sampling leads to a median displacement of $(\Delta x, \Delta y) = (2.5, 39.0)$, the proposed circular sampling leads to a median displacement of $(\Delta x, \Delta y) = (5.0, 4.5)$, which shows that it almost removes the relative prior. The median displacement is not $(0, 0)$, because of the digital nature of an image, the finite number of samples, and the light bias introduced by the intersection of multiple sampling circles.

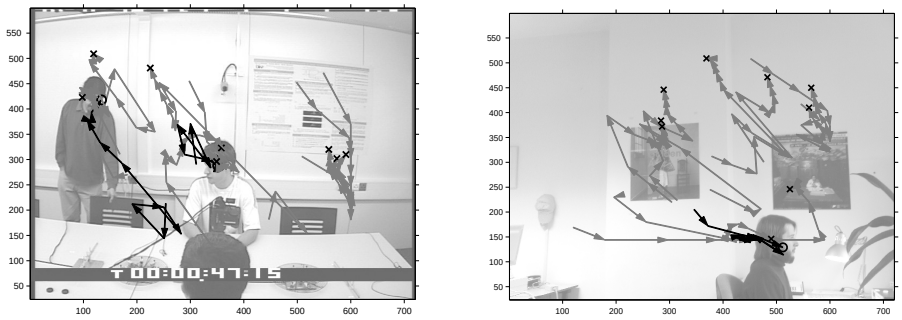


Figure 6.18: Application of ACT-DETECT to images with unusual object locations. We use black arrows for gaze shifts that lead to a classification as object (indicated by a circle), and gray arrows for gaze shifts that lead to a classification as a non-object (indicated by a cross). **Left:** Image from the FGNET set, in which ACT-DETECT finds the face of a standing person. **Right:** Image taken in our office in Maastricht.

[13]We remark that since the face is not labelled, the detection is counted as a false detection (see Section 6.5).

6.6.3 Fine-scale Model

Until now, we have focused on the coarse-scale model. The fine-scale model is evolved for initial gaze locations that are closer to the objects. It exploits object context at a finer scale. The fine-scale model is primarily tuned to properties of the object itself and is useful for determining the exact location of the nearest object. Figure 6.19 shows the actions of the fine-scale model on a grid of 20×20 in the image.

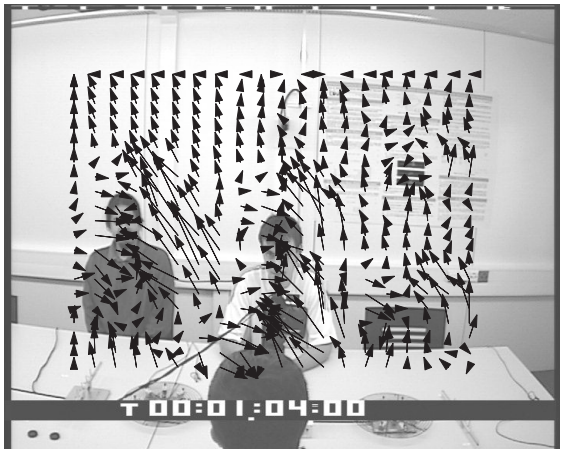


Figure 6.19: Actions taken by the fine-scale model on a 20×20 grid in the image.

From this figure it is clear that ACT-DETECT can approach objects on a finer scale, but that it works best for locations close to the objects. It also leads to the observation that the fine-scale model generally moves slightly up. This may be an indication that the coarse-scale model has a tendency to end up slightly below the faces in the image. Since the fine-scale model is evolved for starting its gaze behaviour at the final positions of the coarse-scale model, it compensates for possible small deviations between these positions and the object locations.

6.6.4 Main Findings of the Analysis

To summarise, the analysis has led to the following four main findings. First, the visual features have been evolved to capture visual properties of the object and its visual context. The feature values contain information on the displacement of the current location to likely object locations. Second, ACT-DETECT maps its visual features to gaze shifts that approach objects in the visual scene. The gaze shifts are partially driven by the information on the location of the closest object, and partially by a non-greedy behaviour that is successful after performing all gaze shifts. Third, the gaze behaviour of ACT-DETECT is quite unsensitive to the prior distribution of object locations in the images (the location prior). Its gaze behaviour

only depends on the prior distribution of relative displacements to the objects in the images (the relative prior). Fourth, the fine-scale model is effective at locations closer to the objects. It compensates for small deviations of the coarse-scale model.

6.7 Setup Car-detection Experiment

The *car-detection experiment* is described in Section 6.7, 6.8, and 6.9. In the experiment, ACT-DETECT is adapted to a car-detection task. In this section, we explain the task and the window-sliding method with which ACT-DETECT is compared. Then, in Section 6.8, the results of the comparison are shown. Subsequently, in Section 6.9 we briefly report on the analysis of the car-detection experiment.

6.7.1 Car-detection Task

The car-detection task concerns the detection of cars from different distances and viewing angles in outdoor images. For the car-detection task, we use the CBCL StreetScenes image set explained in Section 6.1. As mentioned, it contains images of different street scenes, with cars of various sizes. Figure 6.20 shows some example images. We use all 3,546 images in the database and convert them to gray-scale. In our experiment, the order of the images in the image set is first randomised and then one half of the image set is used for training and the other half for testing.

For the car-detection task, ACT-DETECT is provided with $n = 30$ features. Furthermore, the maximal displacement of the gaze window in pixels is $d_{\max} = 640$ for the coarse-scale model and $d_{\max} = 426$ for the fine-scale model. All the other parameter settings are the same as for the face-detection task.

Figure 6.20: Five example images from the CBCL StreetScenes database.

The CBCL image set has been used for object detection in Wolf and Bileschi (2006). However, we cannot compare our detection results with those reported in that article. The reason is that the false positives in Wolf and Bileschi (2006) are expressed as false positive rates, i.e., the proportion of background samples that are classified as object. Since it is not clear how many background samples their method had to consider in an image, we cannot determine the corresponding number of false positives per image. For example, a false positive rate of 0.01 could lead to an average of one false positive per image (in case of 100 samples), but also to an average of 100 false positives (in case of $10,000$ samples). Where one false positive may be a rather good performance, 100 false positives is definitely a bad performance.

Since we cannot use results from the literature, we trained a (Viola and Jones, 2001) object detector for the car-detection task ourselves.

6.7.2 Viola and Jones Object Detector

In order to facilitate reproduction of our results, we employed the openCV-implementation of the Viola and Jones detector, which is freely available on the web[14]. We made two important choices for the training of the detector. First, we allowed the detector to use all integral features present in the openCV library. This means that in addition to the features in Figure 6.8, the detector had access to the diagonal integral features introduced in Lienhart and Maydt (2002). Second, we decided on a size of width $\times$ height $= 35 \times 20$ pixels for the base sample size. This size was based on the ratio between widths and heights of objects in the training set. We used the standard settings of openCV, since preliminary experiments showed that these gave the best results.

6.8 Performance Car-detection Experiment

The evolutionary algorithm also finds successful scanning strategies for the car-detection task. Figure 6.21 shows ten independent runs of the best instance of ACT-DETECT on an example test image. At the end of scanning ($i = 10$) three out of ten runs have reached an object location. This results in the detection of two out of four labelled cars (a recall of 50%).

Figure 6.21: Ten independent runs of ACT-DETECT on an image of the StreetScenes data set.

The performance of ACT-DETECT is compared with that of the openCV implementation of the Viola and Jones detector. Figure 6.22 shows the FROC-plots for ACT-DETECT (dotted line, square markers) and the Viola and Jones detector (solid

[14]OpenCV can be downloaded at http://www.opencv.org/

line, circles). The results on the car-detection task conform to our expectations: the window-sliding method has a better detection performance than ACT-DETECT on the car-detection task for an average number of false positives smaller than four. In the next section, we analyse why this result is different from the face-detection task.

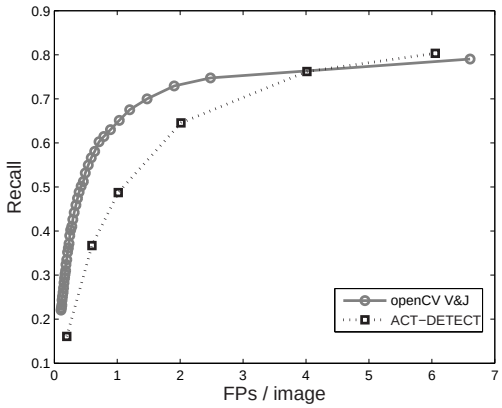


Figure 6.22: FROC-plots of ACT-DETECT (dotted line, square markers) and the openCV implementation of a Viola and Jones detector (solid line, circles).

6.9 Analysis Car-detection Experiment

The analysis of the car-detection experiment is similar to that in Section 6.6. The interested reader can find the entire analysis in Appendix B. In this section, we focus on the difference between the analysis of the car-detection experiment and the face-detection experiment that may explain the difference in the results.

The analysis of the car-detection experiment shows one important difference with the face-detection task: the visual features in the car-detection task capture less information on object locations than the visual features in the face-detection task.

As in Section 6.6, we measured the information contained in clusters of visual inputs on the location of the closest object. We gathered 30 samples for 132 training images by uniformly sampling in the image (resulting in $3,960$ samples). These inputs were clustered with k-means clustering for $k = \langle 2, 3, 4, \ldots, 15 \rangle$. Figure 6.23 shows the relation between the number of clusters and mutual information with the horizontal and vertical displacement $(\Delta X, \Delta Y)$ to the closest object.

We observe that the amount of information in the visual inputs is smaller than for the face-detection task (see Figure 6.14). This may have two possible causes: (1) the visual scene in the car-detection task does not contain sufficient information to locate cars as reliably as in the face-detection task, and (2) the visual features capture less information in the car-detection task than in the face-detection task. The first cause is improbable, since human subjects are able to exploit the information in

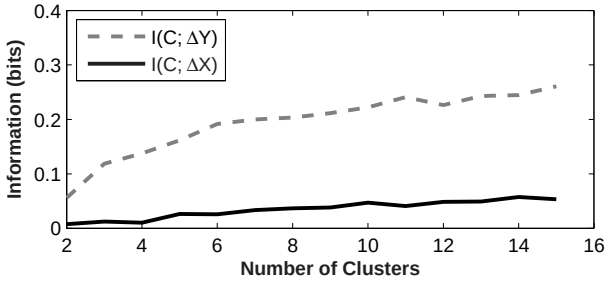


Figure 6.23: The relation between the number of clusters and the mutual information of the clustering with the horizontal distance (ΔX, solid line) and vertical distance (ΔY, dashed line) to the closest object. The mutual information is expressed in bits.

local image samples from the car images to reach object locations (see Section 6.1). The second cause is probable, because the backgrounds in the car-detection experiment have a much larger variation than those in the face-detection experiment. In the case of the car-detection experiment, the visual feature extraction may be too limited to capture sufficient information for outperforming window-sliding methods. Since the information increases with the number of clusters, we cannot rule out that there is a third cause: that the controller is not able to refine the visual input space sufficiently. This may be the case, since the small feedforward neural network used in our experiments is rather limited.

When the detection performance of ACT-DETECT is lower than that of passive scanning methods, the question becomes whether the loss in detection performance is compensated by a sufficient gain in computational efficiency.

6.10 Computational Efficiency

In this section, we focus on RQ 4d: *How computationally efficient is* ACT-DETECT *compared to passive scanning methods?* To answer this question, we make a general comparison of the computational effort of ACT-DETECT and that of window-sliding methods. This comparison is facilitated by the fact that both ACT-DETECT and the window-sliding methods extract features from a local input window. The computational costs C of a window-sliding method (WS) and of ACT-DETECT (AD) can be expressed as follows.

$$C_{\mathrm{WS}} = G_{\mathrm{H}} G_{\mathrm{V}} (F_{\mathrm{WS}} + Cl) + P \tag{6.1}$$

$$C_{\mathrm{AD}} = R(S \times T)(F_{\mathrm{AD}} + Ct) + R(F_{\mathrm{WS}} + Cl) + P \tag{6.2}$$

The variables G_{H} and G_{V} are the number of horizontal and vertical grid points, respectively. Furthermore, F_{WS} is the number of operations necessary for feature extraction in the window-sliding approach, Cl stands for the classifier, and P for preprocessing. For ACT-DETECT, R is the number of independent runs, S the number of scales, and T the number of local samples extracted per scale. F_{AD} is the

number of operations necessary for feature extraction, and Ct for the controller that maps the features to gaze shifts. The cost of ACT-DETECT includes $R(F_{\mathrm{WS}}+Cl)$, since object presence is verified at the final scanning position.

Below, we explain why ACT-DETECT is computationally more efficient than the window-sliding method. The main reason for this is that ACT-DETECT extracts far fewer local samples, i.e., $(R(S \times T) + R) \ll G_{\mathrm{H}}G_{\mathrm{V}}$, while its feature extraction and controller do not take much more computational effort than the feature extraction and classifier of the window-sliding method. For example, in the FGNET-task a window-sliding method that verifies object presence at every point of a grid with a step size of two pixels will extract $335 \times 248 = 83,080$ local samples (based on the image size, the average face size of 50×80 pixels, and the largest step size mentioned in Viola and Jones (2001)). In contrast, with $R = 20$, $S = 2$, and $T = 5$, ACT-DETECT extracts $R(S \times T + 1) = 20 \times 11 = 220$ local samples. Under these conditions, the window-sliding method extracts $83,080/220 \approx 378$ times more local samples than ACT-DETECT[15].

For the experiments in this chapter, we have used integral features. These features are mainly popular because of their fast computation times. Such fast computation times are a necessary property for features used by a window-sliding method, since so many of them have to be extracted during the detection process. In contrast, the sparse sampling of ACT-DETECT opens the doors to other, more costly, input features. These features may better capture the relevant information, and therefore may improve ACT-DETECT's performance.

6.11 Discussion

In this section, we address six challenging questions about ACT-DETECT that a sceptical reader might ask us. These questions concern our implementation choices and the properties and limits of the current active scanning method.

(1) Why did you not use a greedy supervised training algorithm? We have attempted this approach. We made training pairs consisting of both an input feature vector and a target shift to the closest object. Then a greedy supervised training algorithm learned the best possible mapping from input features to scanning vectors. The main disadvantage is that all runs of the model will then shift to the most probable object location in the image. Therefore, the method will be more tuned to precision than to recall.

We have chosen for an evolutionary algorithm, because it can incorporate recall directly in the fitness function and it permits the training of an ensemble of runs. For the same reasons, one could also use a reinforcement learning approach as in Jodogne and Piater (2007)[16].

The disadvantage of using an evolutionary algorithm instead of greedy training is the duration of optimisation: our experiments with evolutionary algorithms take

[15]Note that we did not take object detection at different scales into account for the window-sliding method. The number of scales at which an object can occur would imply a new multiplication factor for the computational costs, which is disadvantageous for the window-sliding method.

[16]See also our motivation for our choice of evolutionary algorithms in Section 3.4.

a few days, where greedy experiments only take half an hour. However, note that training a Viola and Jones classifier is also time-intensive (in our experience it takes more time than our evolutionary algorithms).

(2) How did you determine your parameter values? The experiments we performed in this chapter involve a large number of model and training parameters (the number of scales s, the number of time steps per scale t, the number of instances during evolution λ, etc.). An exhaustive search through these parameters is infeasible because of time-constraints. Therefore, we have made educated guesses in setting most of these parameter values, but cannot guarantee that these settings are optimal.

(3) What happens if the number of objects exceeds the number of independent runs of ACT-DETECT*?* We always employ a fixed number of runs for each image. Therefore, if $R = 10$ runs are performed, ACT-DETECT can only detect 10 objects. There are possibilities to remove this limitation of ACT-DETECT. An elegant solution would be to train a model to detect all objects in one run. However, one could also make additional runs if the number of detections is high. We did not worry much about this limitation of ACT-DETECT, since it does not have its use in problems where we expect a large number of objects. For example, we deem passive scanning methods more suitable for face detection in group pictures.

(4) Can you detect objects at surprising locations? One can interpret a 'surprising object location' in two ways: as an object at an unusual image-coordinate and as an object occurring in an unusual context. ACT-DETECT can detect objects at unusual image coordinates, if its dependence on the prior distribution is not too big (see Subsection 6.6.2). ACT-DETECT can in some cases also cope with unusual appearances of object context, since it is partly driven by object-specific features. In any case, ACT-DETECT will detect fewer objects at surprising locations than window-sliding methods. The opposite side of the coin is that window-sliding methods will have more often false positives in unusual places (such as in trees, cf. Bergboer *et al.*, 2004).

(5) Are varying backgrounds a fundamental problem for active scanning methods? We do not think that the variation of the background in itself is a fundamental problem for active scanning methods. For example, the human subjects in Section 6.1 were able to find cars on the basis of local image samples from a data set with very different backgrounds. The fundamental limit of active scanning depends on the amount of structure in an object's context, and not on how constant it is. If there is such a structure, then its exploitation depends on the visual feature extraction and the controller.

(6) Does the method work on any kind of object-detection task? In principle it can work on any kind of object-detection task. In the worst case there would be no structure in the object's context. Then, the scanning could still be based on the detection of object parts. Clearly, ACT-DETECT works better for some objects and their environments than for others. For example, we believe that it will be well-suited for a miniature robot, which operates within a certain niche and has a fairly stable viewpoint. In contrast, the current implementation of ACT-DETECT may be less suited for finding objects in random images on the web.

6.12 Chapter Conclusion

In this chapter, we introduced a novel method for object detection in natural static images. In contrast to existing object-detection methods, it employs *active scanning*: it uses local image samples to guide the scanning process to object locations in the image. We answered four sub-questions of RQ 4 that are crucial to the success of a gaze control model for object detection.

RQ 4a: *Do local image samples contain information on the locations of objects in the image?* Yes, the experiment in Section 6.1 shows that human subjects are able to map small local image samples to gaze shifts leading towards object locations.

RQ 4b: *How does the detection performance of* ACT-DETECT *compare to that of passive scanning methods?* Our instantiation of an active scanning method, ACT-DETECT, outperforms window-sliding methods on the publicly available FGNET image set, but is outperformed by the openCV implementation of a Viola and Jones detector on the car-detection task. The determining factor for the performance of ACT-DETECT seems to be the variation in the background. Where window-sliding methods are sensitive to large changes of the object appearance, ACT-DETECT seems more sensitive to large changes of the background; ACT-DETECT (with its current feature extraction and controller) performs better on detection tasks with a lower variation in the background.

RQ 4c: *How does* ACT-DETECT *select sensible gaze shifts?* The evolutionary algorithm selects visual features that capture both coarse and detailed contextual features of the objects. ACT-DETECT exploits the information in its visual inputs on the relative positions of objects in the image. However, it does not show a greedy behaviour: it does not always shift its gaze to the most probable location of the closest object. Instead, it shows a non-greedy behaviour that is optimised for the performance reached after multiple gaze shifts. ACT-DETECT's behaviour seems to depend only marginally on the prior distribution of object locations in a data set.

RQ 4d: *How computationally efficient is* ACT-DETECT *compared to passive scanning methods?* ACT-DETECT takes many fewer local samples than a window-sliding method. If the feature extraction represents the main computational cost, active scanning can reduce computational costs in the order of a few hundred times.

On the basis of the answers to the sub-questions, we may answer RQ 4 (*Can an adaptive gaze control model perform on a par with state-of-the-art computer vision models on the task of object detection?*) with a 'yes'. The idea behind active scanning is to trade generality and a limited amount of detection performance against computational efficiency. From the experimental results we may conclude that an active scanning method can be a successful and computationally efficient object-detection method.

The success of ACT-DETECT depends on whether there is a structure in the object's context (as is the case in Figure 6.2), and whether ACT-DETECT's features and

controller allow it to exploit this structure. As mentioned, the current implementation of ACT-DETECT performs better when the variation in the context is limited. We end our chapter by providing an example application for ACT-DETECT and by indicating directions of future research.

Example Application

An example of a task with limited variation in the context is the licence plate detection task studied in de Croon and Postma (2006). It involves a data set provided by Prime Vision B.V.[17]. This company develops licence plate detection applications for speed control. The high number of cars passing by on the highway constrains the computational resources for detecting and reading licence plates in the images. The data set used in the experiments consisted of 2930 labelled gray-scale images containing photos of cars, motor bikes, and trucks on the highway. All photos had been taken from a fixed angle with respect to the highway, but under different lighting conditions. For more details on the experimental setup, we refer to de Croon and Postma (2006). On the data set described, ACT-DETECT achieved a recall of 91.75% with a low false positive rate (on average 0.0825 per image). This performance equals that of a window-sliding method that was applied to the same set (Boom, 2005). Since the OCR-algorithms of Prime Vision B.V. can perform an extremely reliable validation of candidate object locations, ACT-DETECT cannot only reduce computation time but also increase performance. Namely, ACT-DETECT can quickly find the licence plates in the 'easy' images, while other methods can spend more time than before on the remaining 'difficult' images. In this manner, employing ACT-DETECT also leads to a better detection performance.

Directions of Future Research

There are many directions for future research. However, we are particularly interested in two of them.

As a first direction of future research, we want to improve ACT-DETECT so that it is better able to cope with varying backgrounds. We want to investigate more expressive (and computationally more intensive) features and more powerful controllers. In addition, it may be possible to automatically subdivide the training set into clusters of images that have different global appearances. Then, a different model could be trained for each cluster. ACT-DETECT would then activate a different adequate behaviour for each different scene.

As a second direction of future research, we believe that ACT-DETECT might form an improvement to existing object-tracking methods. Actually, ACT-DETECT could be applied to movies as is, extracting a sample and shifting the gaze once per frame. However, we could improve its performance by training it on labelled movies. In that case, it might be able to exploit not only an object's visual context, but also its temporal context.

[17] http://www.primevision.nl/

Chapter

7

General Discussion

In this chapter, we discuss adaptive active vision in the light of our experimental results. The discussion is structured as follows. In Section 7.1, we synthesise our findings on the differences between the probabilistic approach and the adaptive approach to active vision. Then, in Section 7.2, we evaluate the generalisation of the experimental findings in Chapter 4, 5, and 6. In Section 7.3, we focus on the impact of the adaptive approach on computer vision in static images. Finally, in Section 7.4, we discuss the contribution of adaptive active vision models to a better understanding of natural vision systems.

7.1 Probabilistic and Adaptive Active Vision

In Chapter 2 we provided an overview of two active vision approaches: the probabilistic approach and the adaptive approach. The main characteristic of the probabilistic approach is the assumption that active vision is a two-tier process of state estimation and taking actions. In contrast, the essence of the adaptive approach is that it makes no assumptions on what action strategy to follow. Below, we discuss the differences between the approaches in the light of our experimental results.

The probabilistic approach to active vision adopts a clear formal framework to select its actions. The advantage of a formal framework is obvious: it facilitates the proof of certain properties of the models. For example, Denzler and Brown (2002) prove that the belief state of their active vision model converges to the true state, given that the Markov assumption holds and that the state estimator works well. Some assumptions made in the probabilistic approach may be reasonable. For instance, the assumption that variables are Gaussian distributed may be justified by the Central Limit Theorem. However, other assumptions just seem to have as goal to facilitate the calculation of the values necessary for action selection. For example, in many tasks there is no apparent reason why the state space should be Markovian, other than that it simplifies calculations.

Proponents of the probabilistic approach may criticise the adaptive approach, since it does not adopt a clear formal framework for action selection. However, the

absence of such a formalisation allows the adaptive approach to find novel, surprising strategies. These strategies are currently hard to reach under the assumptions of the formal framework of the probabilistic approach. The gaze strategies discovered in our experiments can be considered as a case in point, since they violate the assumptions of the probabilistic approach. Below, we briefly describe two such strategies.

The first strategy is to exploit an *external memory* to solve visual tasks (see Chapter 4 and the work by Nolfi and Marocco, 2002; Spier, 2004; van Dartel *et al.*, 2005). In our experiments, the external memory consisted of the gaze location. Our experiments made clear that the external memory allows the exploitation of multiple observations, despite the memoryless nature of a feedfoward neural network. With the help of the discrete gaze control model, we showed that this implies exploiting the non-Markovian properties of the environment (see Section 4.6.2). Probabilistic active vision models, which are based on the Markov assumption, cannot exploit non-Markovian properties of the environment.

The second strategy is to make gaze shifts that improve the performance without gathering information (see Chapter 5). This strategy is possible in an adaptive active vision model, but would not have been found with a probabilistic approach. A probabilistic model always takes actions in order to gather as much information on the world state as possible.

An intriguing possibility would be to use findings from adaptive active vision models to guide further development of probabilistic active vision models. For example, the formal framework of probabilistic models may be extended as to relax the Markov assumption.

7.2 Generalisation

In Chapter 4, 5, and 6, we have studied how specific adaptive gaze control models handled specific visual tasks. Our analysis of the experimental results revealed that the models employed interesting and useful action strategies. However, the question arises how general these strategies are. In this section, we argue that the action strategies are not based on specific properties of the models and tasks, but on three key characteristics of ACT-FRAME (Chapter 3):

C1. The model has a closed loop of visual inputs and actions.

C2. The visual inputs of the model are local.

C3. The capacity of the model's controller is limited.

We argue that our results mainly depend on these characteristics. Below, we discuss the generalisability of the results for the three gaze control tasks (related to RQ 2, 3, and 4). For each task we describe the task properties that correspond to the key characteristics of ACT-FRAME.

7.2.1 Image Classification

ACT-CLASS' gaze strategy in Chapter 4 arises from the following three properties, related to the three key characteristics. First, ACT-CLASS determines its gaze shifts on the basis of its visual inputs (C1). Second, visual inputs at different image locations convey different amounts of class information to the model (C2). Third, the controller has no memory (C3). As a consequence of these three properties, the model uses its gaze location to increase the information in single observations over time. It does so by moving to better classification areas.

It may be clear that the three properties above do not depend on the details of the model. Variations in the nature of the features or the gaze window will have no effect on the strategy of optimising class information in single observations.

Likewise, we do not expect that the types of strategies employed will depend on the image classification task at hand. For most of these tasks, information can be found at different positions in different images. For example, there is no reason to believe that findings on a gender recognition task would not generalise to an emotion recognition task. In fact, in de Croon *et al.* (2006b), we showed that the findings on the gender recognition task generalise to a task in which the model has to discern happy from angry faces.

Generalisation to multi-class tasks may be more challenging. It may be difficult to exploit an external memory if the model has to recognise many different classes. In addition, a larger number of classes may require a complex behaviour. Although we believe that the closed loop of observations and actions can still be used to facilitate such a task, our models may have to be improved in order to do so.

7.2.2 Control

The eye reflex of ACT-DRIVING in the simulated driving task in Chapter 5 mainly depends on the following three properties. First, there is not only a closed loop between the visual inputs and the gaze shifts, but also between the visual inputs and the car commands (C1). We note that this also implies a closed loop between the gaze shifts and the car commands. Second, visual inputs occur that can disrupt the model's behaviour (C2). Third, the controller has limited capabilities to process the visual inputs (C3). As a consequence of adaptation, the eye controller takes into account the behaviour of the car controller by avoiding disruptive visual inputs.

The three properties above are likely to occur in any control task in which a model with a limited controller encounters many different visual inputs. Since this will be the case for many control tasks, we expect our finding on the simulated driving task to generalise to other control tasks as well: avoiding disruptive inputs is easier than creating an elaborate internal mechanism to handle them correctly.

7.2.3 Object Detection

The strategies of ACT-DETECT in Chapter 6 mainly rely on the following three properties. First, ACT-DETECT has multiple time steps to reach an object location by means of its gaze shifts (C1). Second, the visual features extract information on

probable object locations (C2). Third, the model refines the input space to map the visual inputs to gaze shifts (C3).

As argued in Chapter 6, the gaze strategies for the face-detection task and car-detection task (and their success) are likely to generalise to many other detection tasks.

The non-greedy action selection depends mainly on the first property. The gaze control model is adapted for maximal performance after multiple time steps. Therefore, it can afford to make smaller gaze shifts, taking uncertainty in the visual inputs into account. In addition, it can use its gaze location as an external memory of its previous observations. We have observed the non-greedy behaviour in multiple tasks (see Chapter 6 and also de Croon and Postma, 2007) and expect it to occur in any object detection task in which the detection is evaluated after multiple time steps.

The success of ACT-DETECT seems to depend on the task (the structure in an object's context), the visual features (C2), and on the capacity of the controller to refine the visual input space (C3). Since many objects occur in a structured context, we expect that a possible bottleneck for applying gaze control models to other object detection tasks lies more in the capacities of the gaze control model than in the properties of the task. Our analysis supports this expectation. The amount of information captured by the visual features seemed predictive of the success of the gaze control model relative to window-sliding methods. We suggested that this information is related to the variance in the object context. If the context has many dissimilar appearances, the task becomes more difficult. In this respect, the generalisation to such more complex object-detection tasks may be challenging.

7.3 Computer Vision in Static Images

In this section we focus on computer vision models that handle static images. We briefly discussed the existence of passive and active vision models in the field of computer vision in Chapter 1. Now we elaborate on the discussion of these models on the basis of our experimental results.

The goal of both passive and active computer vision models is to evaluate visual patterns in images. To be able to evaluate such patterns, the models employ one or two steps: an optional first step of searching for pattern candidates, and a second step of evaluating the pattern(s). The difference between passive and active vision models lies in how they perform these steps. Below, we first discuss how passive (Subsection 7.3.1) and active vision models (Subsection 7.3.2) handle static images. Then we discuss what the experimental results have taught us about the advantages and drawbacks of adaptive active vision models (Subsection 7.3.3).

7.3.1 Passive Vision

In passive vision models, the two steps are incorporated as follows. In the first step, passive vision models search for the pattern of interest by using information from the entire image. Most models use an *exhaustive* local search (Burl *et al.*, 1998; Viola

and Jones, 2001), but there are a few models that use globally extracted features to determine the locations of interest (Torralba, 2003; Murphy *et al.*, 2006). We interpret searching here in a broader sense than just object detection: in our interpretation search also includes preprocessing techniques that are supposed to highlight the probable locations of the pattern of interest. An example of this is a passive vision model that detects text in natural images by performing edge detection everywhere in the image (Anthimopoulos, Gatos, and Pratikakis, 2007).

In the second step, the located pattern is automatically evaluated in a task-dependent fashion. Many studies focus solely on this evaluation step. These studies assume either that the pattern has been found successfully by the first step, or that the camera is set up in such a way that the pattern in the images is always in the appropriate place (Zhao and Chellappa, 2002). A straightforward manner to evaluate the pattern, would be to use the raw pixel values of the image (part) containing the pattern. However, a pixel representation may have a very high dimensionality and is therefore unsuitable for learning due to the *curse of dimensionality*. Therefore, computer vision models usually extract features from an image that reduce the dimensionality and / or make the relevant aspects of the pattern more evident. This can be done with the help of dimensionality reduction techniques such as principal component analysis or linear discriminant analysis (for an overview of dimensionality reduction techniques with a focus on computer vision, see van der Maaten, Postma, and van den Herik, submitted), a histogram of local image properties (e.g., Rao, Srihari, and Zhang, 1999), wavelet representations (e.g., Arivazhagan and Ganesan, 2003), filter banks (e.g., Schmid, 2001; Leung and Malik, 2001), etc. We note that wavelet representations and filter banks do not reduce the dimensionality of the pattern representation, but do attempt to make the relevant aspects of the pattern more evident.

7.3.2 Active Vision

Active vision models also proceed in two steps. In the first step, active vision models search for the pattern of interest by using local information from the image. The main difference with the passive vision models is that the active vision models employ an *informed* local search, in which the search locations in the image are iteratively refined. Examples of such models are deformable shape models (Yuille, 1991; Zuo and de With, 2004; Felzenszwalb, 2005), active shape models (Cootes, Edwards, and Taylor, 1999; Keomany and Marcel, 2006; Cristinacce and Cootes, 2007), active appearance models (Edwards, Cootes, and Taylor, 1998; Cootes *et al.*, 1999), active contours (snakes) (Kass *et al.*, 1988; Xu, Ahuja, and Bansal, 2007), and gaze control models that only process a part of the image (e.g., Young *et al.*, 1998; Minut and Mahadevan, 2001).

In the second step, the located pattern can be evaluated in a passive manner, as explained above for the passive vision models. However, the active vision model can also use an informed search to extract the right features from the located pattern, as in Chapter 4.

7.3.3 Adaptive Active Vision

Below, we first discuss the advantages of adaptive active vision models over passive vision models, then the advantages over non-adaptive active vision models, and we end with the drawbacks of adaptive active vision models.

Advantages over Passive Vision Models

Adaptive active vision models have two advantages with respect to passive vision models. First, adaptive active vision models are computationally more efficient than passive vision models in searching for a pattern of interest. This is due to the difference between informed search and exhaustive search: the former is more efficient than the latter. Informed search is successful if the image contains information that can be reliably used for the search. The research in Chapter 6 showed that adaptive active vision models offer the opportunity to achieve considerable gains in computational efficiency for object detection. Their performance on the task depended on the amount of information available for object search, but in general they performed on a par with passive window-sliding methods. Of course, the computational efficiency may be relevant to other tasks than object detection as well.

Second, when evaluating a pattern, adaptive active vision models are less sensitive to the variance in the positions of the constituent pattern parts than passive vision models. The positions of the relevant pattern parts in the image may vary, either due to inaccuracies in the search for the pattern, or due to intrinsic differences between different patterns. The problem that the variance in pattern positions poses to passive vision models, is referred to as the *registration problem* (cf. Rentzeperis *et al.*, 2006). Passive vision models can cope with this problem by employing features that are not sensitive to the variance in pattern part positions, i.e., by using *invariant feature extraction*. Active vision models can use their observations to search for the positions of relevant pattern parts.

The above-mentioned advantages may result in an improvement upon computer vision methods as studied in Moghaddam and Yang (2002), Pnevmatikakis and Polymenakos (2005), and Rentzeperis *et al.* (2006). Rentzeperis *et al.* (2006) study the influence of registration errors (translations of the relevant patterns) on the performances of various face recognition methods. When the face is translated away from the usual location for more than 6% of the distance between the eyes, the performances of all studied methods severely decrease. The model that deals best with these translations is a pseudo 2D Hidden Markov Model (cf. Samaria and Harter, 1994). When the registration error increases from 0% to 6%, the misclassification probability of this method increases from $\sim 6\%$ to $\sim 14\%$. Other methods perform even worse: for the same increase in registration error, the misclassification probability of Linear Discriminant Analysis (cf. Belhumeur, Hespanha, and Kriegman, 1997) increases from 2% to 37%. The results in Chapter 4 show that adaptive active vision models could alleviate the registration problem by searching for the relevant pattern parts in the image. Of course, one can also achieve translation invariance by using the passive method of exhaustively searching for pattern parts. However, this exhaustive search is computationally expensive, as mentioned above.

Interestingly, position variance is not the only type of variance that could be handled by adaptive active vision models. We can aim for handling other types of variances by expanding the possible actions of the gaze control framework in Chapter 3. In this way, it is conceivable that scale invariant recognition could be achieved by introducing the ability to travel in the scale space of the image (zooming in or out on the image). In addition, rotation invariance could be achieved by introducing the ability to rotate the input features. Such manipulations of the input features are not new in themselves. For example, some face detectors can recognise (moderately) rotated faces (e.g., Rowley *et al.*, 1998; Zhoua *et al.*, 2002; Liu *et al.*, 2003, and Huang *et al.*, 2005). Some of these methods (such as Rowley *et al.*, 1998 and Liu *et al.*, 2003) do this by (1) estimating the rotation of the current image sample, (2) rotating the sample so that the estimated rotation is removed, and (3) classifying the resulting sample. In contrast to such models, adaptive active vision models may find novel ways to use rotation actions in the best way possible. The goal for which they use the rotation action may be different from just removing the most probable rotation.

Advantages over Non-adaptive Active Vision Models

Adaptive active vision models have one main advantage over non-adaptive active vision models. The advantage stems from the fact that the action strategies of most non-adaptive active vision models are largely determined in advance. These strategies are likely to be suboptimal, since it is difficult to design a successful closed loop strategy. Designing such a strategy necessitates a type of recursive thinking that can involve many dependencies over multiple time steps. The 'blind' adaptation by, e.g., an evolutionary algorithm, does not have this problem. As a consequence, the advantage of adaptive active vision models is that they exploit the closed loop of actions and observations to a larger extent than non-adaptive active vision models.

This advantage offers an opportunity for many different tasks. Let us take face recognition with a deformable shape model as an example. The deformable shape model was introduced by Yuille (1991), and is often used for face recognition. The recognition method iteratively changes the parameters to deform and move the parts of a face template, so that it best fits an image. The action strategies of most deformable shape models (such as the ones in Yuille, 1991 and Zuo and de With, 2004) are predetermined to always search for the same face parts, such as the hair, eyes, nose, and mouth. The strategies consist of matching a template of these parts in the vicinity of the current search locations. Adaptive active vision models could improve upon such strategies, by making better use of the closed loop of observations and actions. We mention two possible improvements here.

First, the search of an adaptive active vision model may be computationally more efficient than the standard search method. An adaptive model may need only one observation from the image to determine the displacement of the current search location to the target location. This makes exhaustive template matching around the search locations unnecessary. The work in Cristinacce and Cootes (2007) represents a prudent step towards a strategy that does not rely on template matching. It illustrates the computational advantages that can be obtained by estimating the

displacement to the target position at once. However, the method in Cristinacce and Cootes (2007) is limited to displacements of a few pixels around the current search location. In addition, its interaction with the image is restricted to making greedy gaze shifts towards the target location. Adaptive active vision models similar to ACT-DETECT described in Chapter 6 could be even more efficient than the method in Cristinacce and Cootes (2007).

Second, adaptive active vision models go beyond merely handling the translation variance problem. The results in Chapter 4 show that adaptive active vision models do more than just finding the same face parts for each image: they use their observations to determine which parts are important for classifying the current image (see also Lacroix *et al.*, 2007). They may therefore offer an opportunity to search for those face parts that make a person unique. For example, they may recognise one person by his ears, and another person by the many speckles on his face.

Drawbacks

The advantages of the adaptive active vision approach come at a certain cost. The approach has two main drawbacks. The first drawback concerns the difference with passive vision models: not searching the entire image necessarily implies a larger probability of missing the pattern of interest. The success of informed search with respect to exhaustive search depends on how reliable the used information is. This was illustrated in Chapter 6, in which ACT-DETECT performed better on the face-detection task than on the car-detection task. The reason for this was most probably the lower amount of information extracted by the visual features in the car-detection task.

The second drawback concerns both passive vision models and non-adaptive active vision models: more flexibility in the determination of the action strategy leads to a reduced *generality* of the solution. In Machine Learning terms, the adaptive active vision models have a smaller learning bias than other existing models. It is well known that this increases the variance in learning (cf. Mitchell, 1997 and Duda, Hart, and Stork, 2001). In the best case, the loss of generality corresponds to a set of mild restrictions on the images. For example, the gaze control model in Chapter 6 works well in images containing photos of outdoor scenes. It is therefore not able to find a car in a photo that is distorted, such as the one shown in Figure 6.1. In the worst case, the loss of generality results in *over-fitting*. The extra degrees of freedom introduce a larger risk of having a strategy that exploits spurious regularities in the training set of images. As a result, the performance on the training set of images may be much higher than that on the testing set of images. In our experiments we did not notice large differences between training and testing performances, which would be indicative of over-fitting. However, it may be that our training sets were always large enough to prevent over-fitting.

7.4 Natural Vision

To what extent can the adaptive active vision approach contribute to the understanding of natural vision systems? The answer to this question may depend on how 'dynamic' natural vision actually is.

The last decade, Bayesian models of cognition (related to the probabilistic models discussed in Section 7.1) have gained in popularity. Such models have provided parsimonious explanations for many perceptual phenomena (cf. Harris and Wolpert, 1998; Wolpert and Ghahramani, 2000; van Beers and Wolpert, 2002; Knill and Pouget, 2004; Körding and Wolpert, 2004; Stocker and Simoncelli, 2005; Griffiths and Tenenbaum, 2006; Körding and Wolpert, 2006; Nelson and Cottrell, 2007). For example, Weiss, Simoncelli, and Adelson (2002) explained all aspects of a well-known motion illusion with a single Bayesian model, which relied on the assumption that slow movements in the environment are more likely than fast movements. Notably, gaze control models that incorporate elements of the Bayesian view on cognition have been rather successful at predicting the eye movements of human subjects. The model by Torralba *et al.* (2006) achieves a good prediction for a task of finding objects in static images, while the model by (Sprague *et al.*, 2007) realises a good prediction for a (simulated) task of walking on a path, while avoiding obstacles and gathering litter. The core of the success of Bayesian models in explaining cognitive phenomena lies in the optimality of such models (Knill and Pouget, 2004; Nelson and Cottrell, 2007).

Bayesian models are less appropriate to capture dynamical aspects of cognition. There is a considerable amount of work on *dynamic Bayesian networks* (Rabiner, 1989; Rabiner and Juang, 1993; Murphy, 2002; Murphy, 2003), but such networks have two main limitations. First, the relations between variables over time are typically specified on forehand. Second, the relations between variables are limited to a small temporal window, in order to keep learning and inference in the networks tractable (cf. Bishop, 2006, p. 632).

Adaptive models do not have these limitations. First, they do not rely on assumptions regarding the relation between variables over time. Second, adaptive models can exploit relations between variables over many time steps (cf. Hochreiter and Schmidhuber, 1997; Bakker *et al.*, 2003; Tuci, Trianni, and Dorigo, 2004; Gomez and Schmidhuber, 2005; de Croon *et al.*, 2006a; Ampatzis *et al.*, 2008). Hence, adaptive models more naturally incorporate dynamical aspects of cognition (Elman, 1990; Smith *et al.*, 1999; van Gelder, 1999; Beer, 2000; Elman, 2004; Spivey and Dale, 2006). For example, Elman (1990) explains dynamical aspects of word interpretation with a recurrent neural network trained for word prediction. One such a dynamical aspect is that the same word can be interpreted in different manners according to the context in which it occurs.

Concerning vision, adaptive models may be most successful in explaining natural behaviours that are (1) dynamic, and (2) difficult to explain with our current knowledge. We give one example of such a visual behaviour. Land (1988) studies the eye movements and visual apparatus of the copepod *Labidocera*. He provides a plausible explanation for the function of the different sensors in the eye cup. A few of the sensors serve to track a source of light, most probably to stabilise the body

against involuntary disturbances. The remaining sensors form a row and serve to scan across 35° of the dorsal visual field. Presumably, the Labidocera interprets the resulting dynamical inputs to detect conspecifics. The scanning movements are peculiar, since the forward scanning movements are faster than the backward scanning movements. This peculiar scanning behaviour may be best explained with the help of adaptive active vision models.

Many more such visual behaviours exist in nature. If the success of these behaviours depends on the dynamics of the visual inputs, adaptive active vision models may contribute considerably to our understanding of natural vision.

Chapter

8

Conclusion

In this chapter, we first answer our four research questions. Then we address our problem statement and draw a general conclusion. Finally, we indicate directions for future research.

8.1 Answers to the Research Questions

RQ 1: *How does the adaptive active vision approach relate to other approaches to active vision?*

The adaptive active vision approach mainly differs from other approaches, in that it does not adopt a constraining formal framework for action selection. As a consequence, it is more difficult to prove properties of adaptive models. However, adaptive models can discover surprising strategies that are hard to reach under the constraints imposed by a formal framework. In this sense, the strategies of adaptive active vision models could be used to guide the way for, e.g., probabilistic active vision models.

RQ 2: *How does a memoryless adaptive gaze control model handle an image classification task?*

A memoryless adaptive gaze control model handles an image classification task by optimising the class information in a single observation. The model shifts its gaze to good classification locations in the image. It uses its fixation location as an external memory to exploit dependencies between multiple observations and actions. This corresponds to exploiting the non-Markovian properties of the task.

RQ 3: *How does an adaptive gaze control model use its gaze shifts in a control task?*

An adaptive gaze control model uses its gaze shifts to improve the overall performance on a control task. To achieve a good performance, it is necessary to gather information from the environment. However, information gathering is a means to

achieve a good performance, it is not the final goal. The best evolved instance of the model ACT-DRIVING uses its gaze shifts for the following three functions: (1) to find relevant visual features in the visual field, (2) to keep relevant visual features in sight, and (3) to avoid disruptive visual inputs. Where the first two functions concern the gathering of information from the environment, the third function does not. ACT-DRIVING avoids the visual inputs related to obstacles, in order to prevent an overreaction of the car controller.

RQ 4: *Can an adaptive gaze control model perform on a par with state-of-the-art computer vision models on the task of object detection?*

Yes, an adaptive gaze control model can perform on a par with state-of-the-art computer vision models on the task of object detection. Our answer to RQ 4 is based on the answers to the sub-questions RQ 4a–d. To answer the sub-questions, we performed experiments with human subjects and with ACT-DETECT, applying the latter to both a face-detection task and a car-detection task. After evolutionary adaptation, ACT-DETECT has visual features that capture properties of the target object's visual context. It exploits the information in the features to shift its gaze to likely object locations. ACT-DETECT achieves a performance similar to that of window-sliding methods, while extracting many fewer local image samples.

The experimental results suggest that the success of ACT-DETECT depends on whether there is a structure in the object's context, and whether ACT-DETECT's features and controller allow it to exploit this structure. The current implementation of ACT-DETECT performs better when the variation in the context is limited.

8.2 Answer to the Problem Statement

On the basis of the answers to the research questions, we now answer the problem statement.

> **Problem statement**: *How do adaptive active vision models handle challenging visual tasks?*

Adaptive active vision models handle challenging visual tasks by exploiting both their internal processing and their feedback loop with their environment. This feedback loop allows the models to facilitate the execution of their task. They can use the feedback loop to maximise task-specific information in their observations, by using their fixation location as an external memory. They can also use the feedback loop to avoid disruptive visual inputs. This obviates the need for complex internal processing of such inputs.

Did our research contribute to achieving the two general goals mentioned in Chapter 1? The goals are: (1) a better understanding of the general properties of the active vision process, and (2) the improvement of computer vision techniques. We believe that our research forms a step on the way to achieving both these goals.

We have contributed to achieving the first goal by finding interesting possible action strategies. The strategies can in principle be employed by other active vision

systems, including animals and humans. However, whether this is actually the case remains for now an open question.

We have contributed to the second goal by identifying ways in which adaptive active vision can improve computer vision. In particular, adaptive gaze control models form a promise for three reasons: (1) they are computationally efficient, (2) they can successfully handle variance in feature positions, and (3) they can outperform existing active vision models by better exploiting their interaction with the visual scene. The most concrete evidence for the promise that adaptive active vision models hold for the field of computer vision is provided in Chapter 6: ACT-DETECT obtains a good object detection performance with low computational costs.

> **Conclusion:** Based on our research findings, we may conclude that adaptive active vision is a promising approach for discovering the role of actions in vision. It has its restrictions and problems, but making no assumptions on the action strategy allows the models to find novel strategies that can both contribute to a better understanding of the active vision process and to the improvement of computer vision techniques.

8.3 Future Research

In this section, we mention four main directions for future research.

A first direction of future research is to study adaptive active vision models that have an internal state. In the thesis, we have emphasised the possibilities of the models' use of the feedback loop with the environment, but have not studied as much the possibilities of the models' internal state. Adaptive models do not use their internal state as a belief state (cf. Beer, 1995; Beer, 2003; van Dartel *et al.*, 2005; de Croon *et al.*, 2006a). It would be interesting to analyse how adaptive active vision models use their internal state, to find out whether and how they combine observations over multiple time steps. What information do adaptive active vision models extract from the visual inputs over time? Do they consider observations as separate pieces of evidence on the state of the environment, or do they extract a different type of information from multiple observations?

A second direction of future research is to analyse to what extent the use of an external memory generalises to more difficult settings. In Chapter 7, we already discussed the possible limited use of the external memory in the case of image classification with many different classes. However, it would be interesting to study the use of an external memory in different tasks as well.

A third direction of future research is to investigate how we can improve the development of active vision models when they have many degrees of freedom. When the adaptation starts with a random controller, it tends to abstain from using the degrees of freedom that do not directly influence the performance on the task. In the case of the car-driving task, this concerned the eye movements; in many of the evolutionary runs (also those not reported in the thesis) ACT-DRIVING shifts its gaze to one of the bottom corners of the screen. In other words: it abstains from exploiting the degrees of freedom involved in gaze control, and focuses purely on

adapting the parameters for driving. When ACT-DRIVING is at a point in evolution at which it can drive on the basis of the visual inputs in the bottom of the screen, it is difficult to change the gaze behaviour, since it would perturb the driving behaviour. Hence, abstaining from exploiting degrees of freedom at the start of evolution has effects on the use of these degrees of freedom later in evolution.

There are three approaches to facilitating the adaptation of models with many degrees of freedom, which deserve further investigation. The first approach initially freezes and gradually frees degrees of freedom during adaptation (Lungarella and Berthouze, 2002). In this way, all stages of adaptation remain tractable. The second approach would be to incorporate elements in the fitness function that stimulate certain properties of the eye movements. One option is to maximise the mutual information between the visual inputs and the eye movements (cf. Klyubin *et al.*, 2004). This enforces movement of the eyes to different regions in the visual field, without assuming that the model searches for a specific type of information. The third approach would combine adaptation over generations with adaptation during execution. For example, the combination of an evolutionary algorithm with reinforcement learning could bring together the best of both worlds. The evolutionary algorithm could focus on facilitating the adaptation performed by reinforcement learning. As the reinforcement learning method, we could employ the method by Jodogne and Piater (2007). They devised a reinforcement learning method that iteratively refines the visual input space to remove any perceptual ambiguities. Their method is an attempt to make the visual task Markovian, alleviating the problems of the traditional reinforcement methods we studied in de Croon *et al.* (2005c). We believe that one of these approaches should allow us to handle tasks in which the model has many degrees of freedom.

A fourth direction of future research is to study how we can achieve a higher generality with the adaptive active vision models. The limitation in generality is best illustrated by the findings on ACT-DETECT in Chapter 6. The current setup of ACT-DETECT does not suffice to tackle complex problems, involving many different objects and backgrounds. The reason for this is that adding objects and backgrounds to our training set makes the gaze behaviour of ACT-DETECT more general, but less efficient. To retain an efficient behaviour and achieve an augmented generality of application, the adaptive active vision model should be endowed with a mechanism that allows it to select different behaviours for different contexts. One possibility to achieve this is to incorporate an algorithm such as that by Torralba (2003). This algorithm can evaluate the coarse global properties of a visual scene to decide upon a more specific, efficient gaze behaviour.

References

Aloimonos, J. (1990). Purposive and qualitative active vision. *10th International Conference on Pattern Recognition*, Vol. 1, pp. 346–360, Atlantic City, NJ. [4]

Aloimonos, J., Weiss, I., and Bandyopadhyay, A. (1988). Active vision. *International Journal of Computer Vision*, Vol. 1, No. 4, pp. 333–356. [4]

Ampatzis, C., Tuci, E., Trianni, V., and Dorigo, M. (2008). Evolution of signalling in a multi-robot system: categorization and communication. *Adaptive Behavior*, Vol. 16, No. 1, pp. 5–26. [135]

Anstis, S. M. (1973). A chart demonstrating variations in acuity with retinal position. *Vision Research*, Vol. 14, No. 7, pp. 589–592. [1]

Anthimopoulos, M., Gatos, B., and Pratikakis, I. (2007). Multiresolution text detection in video frames. *2nd International Conference on Computer Vision Theory and Applications (VISAPP 2007), Barcelona, Spain* (eds. A. Ranchordas, H. Araújo, and J. Vitrià), pp. 161–166, Institute for Systems and Technologies of Information, Control and Communication (INSTICC). [131]

Arbel, T. and Ferrie, F.P. (2001). Entropy-based gaze planning. *Image and Vision Computing*, Vol. 19, No. 11, pp. 779 – 786. [9, 14, 16]

Arivazhagan, S. and Ganesan, L. (2003). Texture classification using wavelet transform. *Pattern Recognition Letters*, Vol. 24, Nos. 9–10, pp. 1513–1521. [131]

Bäck, T. (1996). *Evolutionary algorithms in theory and practice*. Oxford University Press, Oxford, NY. [14, 21, 42, 51, 75, 105]

Bakker, B., Zhumatiy, V., Gruener, G., and Schmidhuber, J. (2003). A robot that reinforcement-learns to identify and memorize important previous observations. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, Vol. 1, pp. 430–435. [135]

Ballard, D. H. (1991). Animate vision. *Artificial Intelligence*, Vol. 48, No. 1, pp. 57–86. [4, 37, 97]

Baluja, S. and Pomerleau, D. (1997). Dynamic relevance: vision-based focus of attention using artificial neural networks. *Robotics and Autonomous Systems*, Vol. 22, Nos. 3–4, pp. 329–344. [74]

Bandera, C., Vico, F. J., Bravo, J. M., Harmon, M. E., and Barid III, L. C. (1996). Residual Q-learning applied to visual attention. *13th International Conference on Machine Learning (ICML 1996), Bari, Italy* (ed. L. Saitta), pp. 20–27. [39]

Bar, M. (2004). Visual objects in context. *Nature Neuroscience Reviews*, Vol. 5, No. 8, pp. 617 – 625. [95]

Barnes, G. R. and Asselman, P. T. (1991). The mechanism of prediction in human smooth pursuit eye movements. *Journal of Physiology*, Vol. 439, No. 1, pp. 439–461. [1]

Beer, R. D. (1995). On the dynamics of small continuous-time recurrent neural networks. *Adaptive Behavior*, Vol. 3, No. 4, pp. 469–509. [72, 83, 139]

Beer, R.D. (2000). Dynamical approaches to cognitive science. *Trends in cognitive sciences*, Vol. 4, No. 3, pp. 91–99. [135]

Beer, R. D. (2003). The dynamics of active categorical perception in an evolved model agent. *Adaptive Behavior*, Vol. 11, No. 4, pp. 209–243. [10, 72, 139]

Belhumeur, P.N., Hespanha, J.P., and Kriegman, D.J. (1997). Eigenfaces vs. Fisherfaces: recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 7, pp. 711–720. [132]

Bergboer, N. H., Postma, E. O., and van den Herik, H. J. (2004). A context-based model of attention. *16th European Conference on Artificial Intelligence (ECAI 2004), Valencia, Spain* (eds. R. López de Mánteras and L. Saitta), pp. 927–931, IOS Press. [95, 109, 124]

Biederman, I., Glass, A. L., and Stacy, W. Jr. (1973). Searching for objects in real-world scenes. *Journal of Experimental Psychology*, Vol. 97, No. 1, pp. 22–27. [93, 94]

Biederman, I., Mezzanotte, R. J., and Rabinowitz, J. C. (1982). Scene perception: detecting and judging objects undergoing relational violations. *Cognitive Psychology*, Vol. l4, No. 2, pp. l43–l77. [93, 94]

Bishop, C.M. (2006). *Pattern recognition and machine learning*. Springer Science and Business Media, LLC, New York, NY. [135]

Boccignone, G. and Ferraro, M. (2004). Modelling gaze shift as a constrained random walk. *Physica A*, Vol. 331, No. 1, pp. 207–218. [38]

Boom, B. (2005). *Snelle object detectie*. M.Sc. thesis, Vrije Universiteit Amsterdam, Amsterdam, the Netherlands. [126]

Borotschnig, H., Paletta, L., Prantl, M., and Pinz, A. (1999). A comparison of probabilistic, possibilistic and evidence theoretic fusion schemes for active object recognition. *Computing*, Vol. 62, No. 4, pp. 293–319. [13]

Borotschnig, H., Paletta, L., Prantl, M., and Pintz, A. (2000). Appearance-based active object recognition. *Image and Vision Computing*, Vol. 18, No. 9, pp. 715–727. [9, 13, 16, 17, 20, 23]

Brockmann, D. and Geisel, T. (2000). The ecology of gaze shifts. *Neurocomputing*, Vol. 32–33, pp. 643–650. [38]

Brooks, R. A. (1990). Elephants don't play chess. *Robotics and Autonomous Systems*, Vol. 6, Nos. 1–2, pp. 3–15. [3]

Bruce, V. and Young, A. (2000). *In the eye of the beholder*. Oxford University Press, Oxford, NY. [50]

Buchli, J., Righetti, L., and Ijspeert, A.J. (2006). Engineering entrainment and adaptation in limit cycle systems - from biological inspiration to applications in robotics. *Biological Cybernetics*, Vol. 95, No. 6, pp. 645–664. [83]

Burges, C.J.C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, Vol. 2, No. 2, pp. 121–167. [105]

Burl, M. C., Weber, M., and Perona, P. (1998). A probabilistic approach to object recognition using local photometry and global geometry. *5th European Conference on Computer Vision (ECCV 1998), Freiburg, Germany* (eds. H. Burkhardt and B. Neumann), Vol. 2, pp. 628–641, Springer-Verlag. [94, 95, 130]

Calder, A. J., Burton, A. M., Miller, P., Young, A. W., and Akamatsu, S. (2001). A principal component analysis of facial expressions. *Vision research*, Vol. 41, No. 9, pp. 1179–1208. [50]

Carbonetto, P., Dorkó, G., Schmid, C., Kück, H., and de Freitas, N. (in press - online). Learning to recognize objects with little supervision. *International Journal of Computer Vision*. [95]

Chung, M. M. and Jiang, Y. (1998). Contextual cueing: implicit learning and memory of visual context guides spatial attention. *Cognitive Psychology*, Vol. 36, No. 1, pp. 28–71. [94]

Clark, A. (1998). *Being there: putting brain, body, and world together again*. MIT Press, Cambridge, MA. [3]

Cohen, P. (1995). *Empirical methods for artificial intelligence*. MIT Press, Cambridge, MA. [24, 52, 79, 80, 101]

Cootes, T.F., Edwards, G. J., and Taylor, C. J. (1999). Comparing active shape models with active appearance models. *British Machine Vision Conference (BMVC 1999), Nottingham, UK* (eds. T. Pridmore and D. Elliman), Vol. 1, pp. 173–182, British Machine Vision Association. [131]

Cover, T.M. and Thomas, J.A. (1991). *Elements of information theory, Wiley series in telecommunications*. John Wiley and Sons, New York, NY. [162]

Cristinacce, D. and Cootes, T. (2003). A comparison of two real-time face detection methods. 4^{th} *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, Graz, Austria*, pp. 1–8. [109]

Cristinacce, D. and Cootes, T.F. (2007). Boosted regression active shape models. *British Machine Vision Conference (BMVC 2007), Warwick, UK* (eds. N. Rajpoot and A. Bhalerao), Vol. 2, pp. 880–889, British Machine Vision Association. [131, 133, 134]

de Croon, G.C.H.E. (2007). Active object detection. *2nd International Conference on Computer Vision Theory and Applications (VISAPP 2007), Barcelona, Spain* (eds. A. Ranchordas, H. Araújo, and J. Vitrià), pp. 97–103, Institute for Systems and Technologies of Information, Control and Communication (INSTICC). [93, 105, 182]

de Croon, G.C.H.E. and Postma, E. O. (2006). Active object detection. *Belgian-Dutch AI Conference (BNAIC 2006), Namur, Belgium* (eds. P.-Y. Schobbens, W. Vanhoof, and G. Schwanen), pp. 107–114. [93, 103, 126, 182]

de Croon, G.C.H.E. and Postma, E.O. (2007). Sensory-motor coordination in object detection. *1st IEEE Symposium on Artificial Life, Honolulu, HI* (ed. H.A. Abbass), pp. 147–154. [93, 105, 130, 182]

de Croon, G.C.H.E., Postma, E.O., and van den Herik, H.J. (2005a). A situated model of active vision. *Belgian-Dutch AI Conference (BNAIC 2005), Brussels, Belgium* (eds. K. Verbeeck, K. Tuyls, A. Nowé, B. Manderick, and B. Kuijpers), pp. 74–80, Royal Flemish Academy of Belgium for Science and Arts (KVAB). [45, 61, 64, 181]

de Croon, G.C.H.E., Postma, E. O., and van den Herik, H. J. (2005b). Sensory-motor coordination in gaze control. *Applications of Evolutionary Computing (EvoWorkshops 2005), Lausanne, Switzerland* (ed. F. Rothlauf), pp. 334–344, Springer-Verlag. [45, 181]

de Croon, G.C.H.E., van Dartel, M. F., and Postma, E. O. (2005c). Evolutionary learning outperforms reinforcement learning on non-Markovian tasks. *Learning and Memory Workshop of the European Conference on Artificial Life (ECAL 2005), Canterbury, UK.* [42, 140, 181]

de Croon, G.C.H.E., Nolfi, S., and Postma, E.O. (2006a). Towards pro-active embodied agents: on the importance of neural mechanisms suitable to process time information. *Complex Engineered Systems: Science Meets Technology (Understanding Complex Systems)* (eds. D. Braha, A. Minai, and Y. Bar-Yam), pp. 338–363. Springer-Verlag, New York, NY. [72, 135, 139, 181]

de Croon, G.C.H.E., Postma, E.O., and van den Herik, H.J. (2006b). A situated model for sensory-motor coordination in gaze control. *Pattern Recognition Letters*, Vol. 27, No. 11, pp. 1181–1190. [10, 45, 129, 182]

de Croon, G.C.H.E., Sprinkhuizen-Kuyper, I.G., and Postma, E.O. (2006c). Comparing active vision models. Technical Report 06-02, MICC-IKAT, Universiteit Maastricht, Maastricht, the Netherlands. [9, 161, 182]

de Graef, P. C., Christiaens, D., and d'Ydevalle, G. (1990). Perceptual effects of scene context on object identification. *Psychological Research*, Vol. 52, No. 4, pp. 317–329. [93, 94]

Deinzer, F., Denzler, J., and Niemann, H. (2003). Viewpoint selection - planning optimal sequences of views for object recognition. *10th International Conference on Computer Analysis of Images and Patterns (CAIP 2003), Groningen, the Netherlands* (eds. N. Petkov and M.A. Westenberg), pp. 65–73, Springer-Verlag. [9, 13]

Denzler, J. and Brown, C.M. (2000). Optimal selection of camera parameters for state estimation of static systems: an information theoretic approach. Technical report, Computer Science Department, University of Rochester. [11, 13, 161, 162]

Denzler, J. and Brown, C.M. (2002). Information theoretic sensor data selection for active object recognition and state estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 2, pp. 145–157. [4, 9, 11, 13, 16, 17, 20, 23, 32, 127, 161, 162]

Denzler, J., Zobel, M., and Niemann, H. (2003). Information theoretic focal length selection for real-time active 3D object tracking. *9th IEEE International Conference on Computer Vision, Erlangen, Germany*, Vol. 1, pp. 400–407, IEEE Computer Society, Los Alamitos, CA. [13]

Deutsch, B., Zobel, M., Denzler, J., and Niemann, H. (2004). Multi-step entropy based sensors control for visual object tracking. *Pattern Recognition, 26th DAGM Symposium* (eds. C.E. Rasmussen, H.H. Bülthoff, B. Schölkopf, and M.A. Giese), pp. 359–366, Springer-Verlag, Berlin. [9, 13, 30]

Dickmanns, E.D., Mysliwetz, B., and Christians, T. (1990). An integrated spatio-temporal approach to automatic visual guidance of autonomous vehicles. *IEEE transactions on systems, man, and cybernetics*, Vol. 20, No. 6, pp. 1273–1283. [74]

Dorkó, G. and Schmid, C. (2003). Selection of scale-invariant parts for object class recognition. *9th IEEE International Conference on Computer Vision (ICCV 2003)*, Vol. 1, pp. 634–639, IEEE Computer Society, Los Alamitos, CA. [95]

Duda, R.O., Hart, P.E., and Stork, D.G. (2001). *Pattern classification*. John Wiley and Sons, New York, NY. [134]

Easter, S.S., Johns, P.R., and Heckenlively, D. (1974). Horizontal compensatory eye movements in gold fish (Carrassius auratus). *Journal of Comparative Physiology*, Vol. 92, No. 1, pp. 23–35. [1]

Edwards, G.J., Cootes, T.F., and Taylor, C.J. (1998). Face recognition using active appearance models. *5th European Conference on Computer Vision (ECCV 1998), Freiburg, Germany* (eds. H.Burkhardt and B. Neumann), Vol. 2, pp. 581–695, Springer-Verlag. [131]

Elder, J.H., Prince, S.J.D., Hou, Y., Sizintsev, M., and Olevskiy, E. (2007). Pre-attentive and attentive detection of humans in wide-field scenes. *International Journal of Computer Vision*, Vol. 72, No. 1, pp. 47–66. [95]

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, Vol. 14, No. 2, pp. 179–211. [135]

Elman, J.L. (2004). An alternative view of the mental lexicon. *Trends in cognitive sciences*, Vol. 8, No. 7, pp. 301–306. [135]

Emery, N.J. (2000). The eyes have it: the neuroethology, function and evolution of social gaze. *Neuroscience and biobehavioral reviews*, Vol. 24, No. 6, pp. 581–604. [69]

Felzenszwalb, P.F. (2005). Representation and detection of deformable shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 2, pp. 208–220. [131]

Fergus, R., Perona, P., and Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. *Computer Vision and Pattern Recognition (CVPR 2003), Madison, WI*, Vol. 2, pp. 264–271, IEEE Computer Society. [95]

Fergus, R., Perona, P., and Zisserman, A. (2007). Weakly supervised scale-invariant learning of models for visual recognition. *International Journal of Computer Vision*, Vol. 71, No. 3, pp. 273–303. [95]

Findlay, J.M. and Gilchrist, I.D. (2003). *Active vision: the psychology of looking and seeing (Oxford psychology series)*. Oxford University Press, New York, NY. [5]

Floreano, D., Kato, T., Marocco, D., and Sauser, E. (2004). Coevolution of active vision and feature selection. *Biological Cybernetics*, Vol. 90, No. 3, pp. 218–228. [4, 5, 10, 40, 42, 46, 70, 71, 72, 97]

Floreano, D., Dürr, P., and Mattiussi, C. (in press). Neuroevolution: from architectures to learning. *Evolutionary Intelligence*. [42]

Fröba, B. and Küllbeck, C. (2002). Robust face detection at video frame rate based on edge orientation features. *5th IEEE international conference on automatic face and gesture recognition 2002*, pp. 342–347, IEEE Computer Society, Los Alamitos, CA. [109]

Funahashi, K. and Nakamura, Y. (1993). Approximation of dynamical systems by continuous time recurrent neural networks. *Neural Networks*, Vol. 6, No. 6, pp. 801–806. [72, 83]

Geusebroek, J.M., Burghouts, G.J., and Smeulders, A.W.M. (2005). The Amsterdam library of object images. *International Journal of Computer Vision*, Vol. 61, No. 1, pp. 103–112. [16]

Gibson, J. J. (1979). *The ecological approach to visual perception*. Houghton Mifflin, Boston, MA. [3]

Gomez, F.J. and Schmidhuber, J. (2005). Co-evolving recurrent neurons learn deep memory POMDPs. *Genetic and Evolutionary Computation Conference (GECCO 2005), Washington, DC* (eds. H.-G. Beyer and U.-M. O'Reilly), pp. 491 – 498, ACM Press. [43, 135]

Griffiths, T.L. and Tenenbaum, J.B. (2006). Optimal predictions in everyday cognition. *Psychological science*, Vol. 17, No. 9, pp. 767 – 773. [135]

Harris, C.M. and Wolpert, D.M. (1998). Signal-dependent noise determines motor planning. *Nature*, Vol. 394, pp. 780 – 784. [135]

Harvey, I., Husbands, P., and Cliff, D. (1994). Seeing the light: artificial evolution, real vision. Technical Report CSRP 317, School of Cognitive and Computing Sciences, University of Sussex. [5, 10, 40]

Hayhoe, M. and Ballard, D. (2005). Eye movements in natural behavior. *TRENDS in Cognitive Sciences*, Vol. 9, No. 4, pp. 188–193. [2]

Henderson, J. M. (2003). Human gaze control during real-world scene perception. *TRENDS in Cognitive Sciences*, Vol. 7, No. 11, pp. 498–504. [2]

Henderson, J. M., Jr., P. A. Weeks, and Hollingworth, A. (1999). Effects of semantic consistency on eye movements during scene viewing. *Journal of Experimental Psychology: Human Perception and Performance*, Vol. 25, No. 1, pp. 210–228. [93, 94]

Henderson, J.M., Falk, R., Minut, S., Dyer, F.C., and Mahadevan, S. (2001). Gaze control for face learning and recognition by humans and machines. *From fragments to objects: segmentation processes in vision* (eds. T. Shipley and P. Kellman), pp. 463–481, Elsevier, New York, NY. [3, 39]

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, Vol. 9, No. 8, pp. 1735–1780. [135]

Hoiem, D., Efros, A. A., and Hebert, M. (2005). Geometric context from a single image. *10th IEEE International Conference on Computer Vision (ICCV 2005), Bejing, China* (eds. S. Ma and H.-Y. Shum), Vol. 1, pp. 654–661, IEEE Computer Society, Washington, DC. [95]

Hoiem, D., Efros, A. A., and Hebert, M. (2006). Putting objects in perspective. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2006), New York, NY*, Vol. 2, pp. 2137–2144, IEEE Computer Society, Los Alamitos, CA. [95]

Holland, J. (1992). Genetic Algorithms. *Scientific American*, pp. 66–72. [14, 42]

Holub, A. D., Welling, M., and Perona, P. (2005). Combining generative models and Fisher kernels for object recognition. *10th IEEE International Conference on Computer Vision (ICCV 2005), Bejing, China* (eds. S. Ma and H.-Y. Shum), pp. 136 – 143, IEEE Computer Society, Washington, DC. [95]

Hornby, G.S., Takamura, S., Yokono, J., Hanagata, O., Fujita, M., and Pollack, J. (2000). Evolution of controllers from a high-level simulator to a high DOF robot. *3rd Internation Conference on Evolvable Systems: from Biology to Hardware (ICES 2000), Edinburgh, UK* (ed. J. Miller), pp. 80–89, Springer-Verlag, London, UK. [10]

Huang, C., Ai, H., Li, Y., and Lao, S. (2005). Vector boosting for rotation invariant multi-view face detection. *10th IEEE International Conference on Computer Vision (ICCV 2005), Bejing, China* (eds. S. Ma and H.-Y. Shum), Vol. 1, pp. 446–453, IEEE Computer Society, Washington, DC. [133]

Husbands, P., Harvey, I., Cliff, D., and Miller, G. (1997). Artificial evolution: a new path for artificial intelligence? *Brain and Cognition*, Vol. 34, No. 1, pp. 130–159. [4]

Ijspeert, A.J. and Crespi, A. (2007). Online trajectory generation in an amphibious snake robot using a lamprey-like central pattern generator model. *IEEE International Conference on Robotics and Automation (ICRA 2007), Rome, Italy*, pp. 262–268, IEEE Computer Society. [83]

Itti, L., Koch, C., and Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 11, pp. 1254 – 1259. [38]

Jain, A.K., Murthy, M.N., and Flynn, P.J. (1999). Data clustering: a review. *ACM Computing Surveys*, Vol. 31, No. 3, pp. 264–323. [60]

Jodogne, S. and Piater, J. (2007). Closed-loop learning of visual control policies. *Journal of Artificial Intelligence Research*, Vol. 28, pp. 349–391. [123, 140]

Kadir, T. and Brady, M. (2001). Scale, saliency and image description. *International Journal of Computer Vision*, Vol. 45, No. 2, pp. 83–105. [95]

Kampe, K.K.W., Frith, C.D., Dolan, R.J., and Frith, U. (2001). Reward value of attractiveness and gaze. *Nature*, Vol. 413, p. 589. [69]

Kass, Michael, Witkin, Andrew, and Terzopoulos, Demetri (1988). Snakes: active contour models. *International Journal of Computer Vision*, Vol. 1, No. 4, pp. 259–268. [9, 131]

Kato, T. and Floreano, D. (2001). An evolutionary active-vision system. *Congress on Evolutionary Computation (CEC 2001), Seoul, South Korea*, Vol. 1, pp. 107–114, IEEE Computer Society. [5, 40, 97]

Keomany, J. and Marcel, S. (2006). Active shape models using local binary patterns, IDIAP-RR 07. Technical report, IDIAP. [131]

Kim, T.-K. and Kittler, J. (2005). Composite support vector machines with extended discriminative features for accurate face detection. *IEICE Transactions on Information and Systems*, Vol. E88-D, No. 10, pp. 2373–2379. [105]

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, Vol. 220, No. 4598, pp. 671–680. [42]

Kirsh, D. and Maglio, P. (1994). On distinguishing epistemic from pragmatic action. *Cognitive Science*, Vol. 18, No. 4, pp. 513–549. [7]

Klarquist, W. N. and Bovik, A. C. (1998). FOVEA: a foveated vergent active stereo vision system for dynamic three-dimensional scene recovery. *IEEE Transactions on Robotics and Automation*, Vol. 14, No. 5, pp. 755–770. [39]

Klyubin, A.S., Polani, D., and Nehaniv, C.L. (2004). Organization of the information flow in the perception-action loop of evolved agents. *2004 NASA/DoD Conference on Evolvable Hardware (EH 2004), Seattle, WA* (eds. R.S. Zebulum, D. Gwaltney, G. Hornby, D. Keymeulen, J. Lohn, and A. Stoica), pp. 177–180, IEEE Computer Society. [63, 140]

Knill, D.C. and Pouget, A. (2004). The Bayesian brain: the role of uncertainty in neural coding and computation. *Trends in Neurosciences*, Vol. 27, No. 12, pp. 712–719. [135]

Körding, K.P. and Wolpert, D.M. (2004). Bayesian integration in sensorimotor learning. *Nature*, Vol. 427, pp. 244 – 247. [135]

Körding, K.P. and Wolpert, D.M. (2006). Bayesian decision theory in sensorimotor control. *Trends in cognitive science*, Vol. 10, No. 7, pp. 319–326. [135]

Kortmann, R. (2001). Embodied cognitive science. *Robo Sapiens - the first Dutch symposium on embodied intelligence, Utrecht, The Netherlands* (eds. W. de Back, T. van der Zant, and L. Zwanepol), Vol. 24. [3]

Kranczioch, C., Debener, S., Schwarzbach, J., Goebel, R., and Engel, A. K. (2005). Neural correlates of conscious perception in the attentional blink. *Neuroimage*, Vol. 24, No. 3, pp. 704–714. [2]

Krauzlis, R. J. (2005). The control of voluntary eye movements: new perspectives. *The neuroscientist*, Vol. 11, No. 2, pp. 124–137. [1]

Kröse, B.J.A. and Bunschoten, R. (1999). Probabilistic localization by appearance models and active vision. *IEEE International Conference on Robotics and Automation (ICRA 1999), Detroit, MI*, Vol. 3, pp. 2255–2260, IEEE Computer Society. [9, 14, 20]

Kruppa, H., Castrillon-Santana, M., and Schiele, B. (2003). Fast and robust face finding via local context. *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS 2003), Nice, France*, pp. 157–164, IEEE Computer Society. [95, 109, 110]

Lacroix, J.P.W., Postma, E.O., and van den Herik, H.J. (2007). Modeling visual classification using bottom-up and top-down fixation selection. *29th annual conference of the Cognitive Science Society (CogSci 2007), Nashville, TN*, pp. 419–424. [39, 134]

Laeng, B. and Teodorescu, D.S. (2002). Eye scanpaths during visual imagery reenact those of perception of the same visual scene. *Cognitive Science*, Vol. 26, No. 2, pp. 207–231. [69]

Land, M.F. (1988). The functions of eye and body movements in Labidocera and other copepods. *Journal of Experimental Biology*, Vol. 140, No. 1, pp. 381–391. [135]

Land, M.F. (1992). Predictable eye head coordination during driving. *Nature*, Vol. 359, pp. 318–320. [70]

Land, M. F. and Hayhoe, M. (2001). In what ways do eye movements contribute to everyday activities? *Vision Research*, Vol. 41, Nos. 25–26, pp. 3559–3565. [1, 2]

Land, M.F. and Lee, D.N. (1994). Where we look when we steer. *Nature*, Vol. 369, pp. 742–744. [70]

Land, M. F. and Nilsson, D.-E. (2002). *Animal eyes*. Oxford University Press, New York, NY. [1]

Lee, T.S. and Yu, S. X. (1999). An information-theoretic framework for understanding saccadic eye movements. *Advances in Neural Information Processing Systems 12 (NIPS 1999), Denver, CO* (eds. S.A. Solla, T.K. Leen, and K.-R. Mller), pp. 834–840, MIT Press. [39, 69]

Leung, T. and Malik, J. (2001). Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, Vol. 43, No. 1, pp. 29–44. [131]

Lienhart, R. and Maydt, J. (2002). An extended set of Haar-like features for rapid object detection. *International Conference on Image Processing (ICIP 2002), Rochester, NY*, Vol. 1, pp. 900–903, IEEE Computer Society. [120]

Liu, H., Yan, S., Chen, X., and Gao, W. (2003). Rotated face detection in color images using radial template (RT). *International Conference on Multimedia and Expo (ICME 2003), Baltimore, MD*, Vol. 3, pp. 137–140, IEEE Computer Society, Los Alamitos, CA. [133]

Li, S. Z. and Zhang, Z. (2004). Floatboost learning and statistical face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 9, pp. 1112–1123. [95]

Loeff, N., Arora, H., Sorokin, A., and Forsyth, D. (2005). Efficient unsupervised learning for localization and detection in object categories. *Advances in Neural Information Processing Systems (NIPS 2005), Vancouver, Canada* (eds. Y. Weiss, B. Schölkopf, and J. Platt), pp. 811–818, MIT Press. [95]

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, Vol. 60, No. 2, pp. 91–110. [95]

Lungarella, M. and Berthouze, L. (2002). Adaptivity via alternate freeing and freezing of degrees of freedom. *9th International Conference on Neural Information Processing (ICONIP 2002), Orchid Country Club, Singapore* (eds. L. Wang, J.C. Rajapakse, K. Fukushima, S.-Y. Lee, and X. Yao), Vol. 1, pp. 482 – 487. [140]

Macinnes, I. and Di Paolo, E.A. (2006). The advantages of evolving perceptual cues. *Adaptive Behavior*, Vol. 14, No. 2, pp. 147–156. [4]

Marocco, D. and Floreano, D. (2002). Active vision and feature selection in evolutionary behavioral systems. *7th International Conference on Simulation of Adaptive Behavior, from Animals to Animats (SAB 2002), Edinburgh, UK*, pp. 247–255, MIT Press, Cambridge, MA. [5, 40, 97]

Marr, D. (1982). *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Freeman, New York, NY. [3, 4]

Mattiussi, C. and Floreano, D. (2004). Evolution of analog networks using local string alignment on highly reorganizable genomes. *NASA/DoD Conference on Evolvable Hardware (EH 2004), Seattle, WA* (eds. R.S. Zebulum, D. Gwaltney, G. Hornby, D. Keymeulen, J. Lohn, and A. Stoica), pp. 30–37, IEEE Computer Society. [42]

Mikolajczyk, K., Leibe, B., and Schiele, B. (2006). Multiple object class detection with a generative model. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06), New York, NY*, pp. 26–36, IEEE Computer Society. [95]

Minut, S. and Mahadevan, S. (2001). A reinforcement learning model of selective visual attention. *5th International Conference on Autonomous agents (Agents 2001), Montreal, Canada*, pp. 457–464, ACM Press, New York, NY. [9, 39, 97, 131]

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill Companies, Inc., New York, NY. [134]

Mitri, S., Frintrop, S., Pervlz, K., Surmann, H., and Nchter, A. (2005). Robust object detection at regions of interest with an application in ball recognition. *IEEE 2005 International Conference Robotics and Automation (ICRA 2005), Barcelona, Spain*, pp. 126–131, IEEE Computer Society. [95]

Moghaddam, B. and Pentland, A. (1997). Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Vision*, Vol. 19, No. 7, pp. 696–710. [94]

Moghaddam, B. and Yang, M.-H. (2002). Learning gender with support faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 5, pp. 707–711. [50, 132]

Murphy, K.P. (2002). *Dynamic Bayesian Networks: Representation, Inference and Learning*. Ph.D. thesis, University of California, Berkeley, CA. [135]

Murphy, K.P. (2003). Dynamic Bayesian networks. *Probabilistic Graphical Models* (ed. M. Jordan). [135]

Murphy, K., Torralba, A., Eaton, D., and Freeman, W.T. (2006). Object detection and localization using local and global features. *Toward Category-Level Object Recognition (Sicily Workshop on Object Recognition)* (eds. J. Ponce, M. Hebert, C. Schmid, and A. Zisserman), pp. 382 – 400, Springer Berlin / Heidelberg. [96, 131]

Neider, M. B. and Zelinski, G. J. (2006). Scene context guides eye movements during visual search. *Vision Research*, Vol. 46, No. 5, pp. 614–621. [94]

Nelson, J.D. and Cottrell, G.W. (2007). A probabilistic model of eye movements in concept formation. *Neurocomputing*, Vol. 70, Nos. 13–15, pp. 2256–2272. [135]

Nolfi, S. (2002). Power and the limits of reactive agents. *Neurocomputing*, Vol. 42, Nos. 1–4, pp. 119–145. [45]

Nolfi, S. and Floreano, D. (2000). *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. MIT Press / Bradford Books, Cambridge, MA. [4, 43]

Nolfi, S. and Marocco, D. (2002). Evolving robots able to visually discriminate between objects with different size. *International Journal of Robotics and Automation*, Vol. 17, No. 4, pp. 163–170. [4, 9, 10, 54, 67, 91, 128]

Oliva, A., Torralba, A., Castelhano, M. S., and Henderson, J. M. (2003). Top-down control of visual attention in object detection. *10th IEEE International Conference on Image Processing (ICIP 2003), Barcelona, Spain*, Vol. 1, pp. 253–256, IEEE Computer Society. [94]

Osuna, E., Freund, R., and Girosi, F. (1997). Training support vector machines: an application to face detection. *Computer Vision and Pattern Recognition (CVPR 1997), San Juan, Puerto Rico*, pp. 130–136, IEEE Computer Society, Los Alamitos, CA. [95, 105]

Paletta, L., Prantl, M., and Pinz, A. (1998). Reinforcement learning for autonomous three-dimensional object recognition. *6th Symposium on Intelligent Robotics Systems (SIRS 1998), Edinburgh, UK*, pp. 63–72. [9, 13, 16, 17, 23, 32]

Paletta, L., Fritz, G., and Seifert, C. (2005). Q-Learning of sequential attention for visual object recognition from informative local descriptors. *22nd International Conference on Machine Learning (ICML 2005), Bonn, Germany* (eds. L. de Raedt and S. Wrobel), pp. 649–656, ACM Press. [39]

Palmer, S. T. (1975). The effects of contextual scenes on the identification of objects. *Memory and Cognition*, Vol. 3, No. 5, pp. 519–526. [94]

Papageorgiou, C. and Poggio, T. (2000). A trainable system for object detection. *International Journal of Computer Vision*, Vol. 38, No. 1, pp. 15–33. [94]

Partridge, M. and Calvo, R.A. (1998). Fast dimensionality reduction and simple PCA. *Intelligent Data Analysis*, Vol. 2, Nos. 1–4, pp. 203–214. [20]

Paul, H., Nalbach, H.-O., and Varjú, D. (1990). Eye movements in the rock crab Pachygrapsus marmoratus walking along straight and curved paths. *Journal of Experimental Biology*, Vol. 154, No. 1, pp. 81–97. [1]

Perko, R. and Leonardis, A. (2007). Learning visual context for object detection. *16th IEEE Electrotechnical and Computer Science Conference (ERK 2007), Portorož, Slovenia*, Vol. B, pp. 163–166, IEEE Computer Society. [95, 115]

Peters, R. J. and Itti, L. (2007). Beyond bottom-up: incorporating task-dependent influences into a computational model of spatial attention. *12th IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007), Minneapolis, MN*, pp. 1–8, IEEE Computer Society. [38]

Pfeifer, R. and Scheier, C. (1999). *Understanding Intelligence*. MIT Press, Cambridge, MA. [3, 45, 46]

Platt, J. (1999). Using sparseness and analytic QP to speed training of support vector machines. *Advances in Neural Information Processing Systems (NIPS 1999), Denver, CO* (eds. M.S. Kearns, S.A. Solla, and D.A. Cohn), Vol. 11, pp. 557–563, MIT Press. [105]

Pnevmatikakis, A. and Polymenakos, L. (2005). An automatic face detection and recognition system for video streams. *2nd Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms (MLMI 2005), Edinburgh, UK*, Springer-Verlag. [132]

Poel, M., van Breemen, A., Nijholt, A., Heylen, D.K., and Meulemans, M. (2007). Gaze behavior, believability, likability and the iCat. *6th Workshop on Social Intelligence Design, Enschede, the Netherlands* (eds. A. Nijholt, O. Stock, and T. Nishida), CTIT Workshop Proceedings Series, pp. 109–124, Universiteit Twente, Enschede. [69]

Pomerleau, D. (1995). Neural network vision for robot driving. *The Handbook of Brain Theory and Neural Networks* (ed. M. Arbib). [74]

Privitera, C. M. and Stark, L. W. (2000). Algorithms for defining visual regions-of-interest: comparison with eye fixations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 9, pp. 970–982. [38, 39]

Qi, Y., Doermann, D., and DeMenthon, D. (2001). Hybrid independent component analysis and support vector machine learning scheme for face detection. *IEEE*

International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2001), Salt Lake City, UT, Vol. 3, pp. 1481–1484, IEEE Computer Society, Los Alamitos, CA. [105]

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, Vol. 77, No. 2, pp. 257–285. [135]

Rabiner, L. and Juang, B.H. (1993). *Fundamentals of speech recognition*. Prentice Hall. [135]

Rajashekar, U., Cormack, L.K., and Bovik, A.C. (2003). Image features that draw fixations. *10th IEEE International Conference on Image Processing (ICIP 2003), Barcelona, Spain*, Vol. 3, pp. 313–316, IEEE Computer Society. [2, 38]

Rajashekar, U., van der Linde, I., Bovik, A.C., and Cormack, L.K. (2007). Foveated analysis of image features at fixations. *Vision Research*, Vol. 47, No. 25, pp. 3160–3172. [2]

Rao, R. P. N., Zelinsky, G. J., Hayhoe, M. M., and Ballard, D. H. (1995). Modeling saccadic targeting in visual search. *Advances in Neural Information Processing Systems (NIPS 1995)* (eds. D. Touretzky, M. Mozer, and M. Hasselmo), Vol. 8, pp. 830–836, MIT Press. [1, 39, 94, 97]

Rao, R. P. N., Zelinsky, G. J., Hayhoe, M. M., and Ballard, D. H. (1997). Eye movements in visual cognition: a computational study. Technical Report 97.1, Department of Computer Science, University of Rochester. [39, 97]

Rao, A., Srihari, R.K., and Zhang, Z. (1999). Spatial color histograms for content-based image retrieval. *11th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 1999), Chicago, IL*, pp. 183–186, IEEE Computer Society. [131]

Rasolzadeh, B., Björkman, M., Hayman, E., and Eklundh, J. (2006). An attentional system combining top-down and bottom-up influences. *International Cognitive Vision Workshop, in conjunction with ECCV 2006, Graz, Austria.* [38]

Rentzeperis, E., Stergiou, A., Pnevmatikakis, A., and Polymenakos, L. (2006). Impact of face registration errors on recognition. *IFIP International Federation for Information Processing: Artificial Intelligence Applications and Innovations*, pp. 187–194, Springer Boston. [132]

Richardson, D.C., Matlock, T., Crosby, J.R., and Dale, R. (2006). Figurative, spontaneous, interactive and potentially offensive: three projects with rich visual and linguistic stimuli. *Cognitive Science 2006 Workshop: What have eye movements told us so far, and what is next?* [69]

Rivlin, E. and Aloimonos, Y. (1992). Purposive active vision: combining perception and action. *International AI Symposium.* [4]

Roberts, J. F., Stirling, T., Zufferey, J.-C., and Floreano, D. (2007). Quadrotor using minimal sensing for autonomous indoor flight. *3rd US-European Competition and Workshop on Micro Air Vehicle Systems (EMAV 2007), Toulouse, France*. [97]

Rowley, H. A., Baluja, S., and Kanade, T. (1998). Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 1, pp. 23–38. [94, 95, 133]

Rubinstein, R.Y. and Kroese, D.P. (2004). *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning*. Springer-Verlag, New York. [42]

Rybak, I.A., Gusakova, V.I., Golovan, A.V., Podladchikova, L.N., and Shevtsova, N.A. (1998). A model of attention-guided visual perception and recognition. *Vision Research*, Vol. 38, Nos. 15–16, pp. 2387–2400. [39]

Salah, A. A., Alpaydin, E., and Akarun, L. (2001). A selective attention based method for visual pattern recognition. *23rd Annual Conference of Cognitive Science Society, Edinburgh, UK* (eds. J. D. Moore and K. Stenning), pp. 881–886. [39]

Samaria, F. and Harter, A. (1994). Parametrisation of a stochastic model for human face identification. *2nd IEEE Workshop on Applications of Computer Vision, Sarasota, FL*, pp. 138–142, IEEE Computer Society. [132]

Scheier, C., Pfeifer, R., and Kuniyoshi, Y. (1998). Embedded neural networks: exploiting constraints. *Neural Networks*, Vol. 11, Nos. 7–8, pp. 1551–1569. [4, 54]

Schiele, B. and Crowley, J.L. (1997). Transinformation of object recognition and its application to viewpoint planning. *Robotics and Autonomous Systems*, Vol. 21, No. 1, pp. 95–106. [20]

Schiele, B. and Crowley, J.L. (1998). Transinformation for active object recognition. *6th International Conference on Computer Vision (ICCV 1998), Bombay, India*, pp. 249–255, IEEE Computer Society, Washington DC, USA. [9, 14]

Schiele, B. and Crowley, J. L. (2000). Recognition without correspondance using multidimensional receptive field histograms. *International Journal of Computer Vision*, Vol. 36, No. 1, pp. 31–50. [94]

Schilstra, C. and van Hateren, J. H. (1998). Stabilizing gaze in flying blowflies. *Nature*, Vol. 395, p. 654. [1]

Schlesinger, M. (2003). A Lesson from Robotics: Modeling Infants as Autonomous Agents. *Adaptive Behavior*, Vol. 11, No. 2, pp. 97–107. [40]

Schlesinger, M. and Casey, P. (2003). Where infants look when impossible things happen: simulating and testing a gaze-direction model. *Connection Science*, Vol. 15, No. 4, pp. 271–280. [40]

Schlesinger, M. and Parisi, D. (2001). The Agent-Based Approach: A New Direction for Computational Models of Development. *Developmental review*, Vol. 21, pp. 121–146. [40]

Schmid, C. (2001). Constructing models for content-based image retrieval. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2001), Kauai, HI*, Vol. 2, pp. 39–45, IEEE Computer Society. [131]

Schmidhuber, J. (1990). Reinforcement learning in Markovian and non-Markovian environments. *Advances in neural information processing systems (NIPS 1990), Denver, CO* (eds. D.S. Lippman, J.E. Moody, and D.S. Touretzky), pp. 500–506, Morgan Kaufmann Publishers Inc., San Francisco, CA. [5, 40, 97]

Schneiderman, H. and Kanade, T. (2000). A statistical approach to 3D object detection applied to faces and cars. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2000), Hilton Head Island, SC*, Vol. 1, pp. 746–751, IEEE Computer Society. [95]

Sela, G. and Levine, M. D. (1997). Real-time attention for robotic vision. *Real-Time Imaging*, Vol. 3, No. 3, pp. 173–194. [39]

Shannon, C.E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, Vol. 27, pp. 379–423, 623–656. [54, 60, 62, 112]

Smeraldi, F., Capdevielle, N., and Bigün, J. (1999). Facial features detection by saccadic exploration of the Gabor decomposition and support vector machines. *11th Scandinavian Conference on Image Analysis (SCI 1999), Kangerlussuaq, Greenland*, Vol. 1, pp. 39–44. [39, 97]

Smith, L.B., Thelen, E., Titzer, R., and McLin, D. (1999). Knowing in the context of acting: the task dynamics of the A-not-B error. *Psychological Review*, Vol. 106, No. 2, pp. 235–260. [135]

Spier, E. (2004). Behavioural categorisation: behaviour makes up for bad vision. *9th Interational Conference on the Simulation and Synthesis of Life (Artificial Life IX), Boston, MA* (eds. J. Pollack, M. Bedau, P. Husbands, T. Ikegami, and R. Watson), pp. 133–138, MIT Press. [128]

Spivey, M.J. and Dale, R. (2006). Continuous dynamics in real-time cognition. *Current directions in psychological science*, Vol. 15, No. 5, pp. 207–211. [135]

Sprague, N., Ballard, D., and Robinson, A. (2007). Modeling embodied visual behaviors. *ACM Transactions on Applied Perception*, Vol. 4, No. 2, p. 11. [39, 135]

Stanley, K.O. and Miikkulainen, R. (2004). Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research*, Vol. 21, pp. 63–100. [42]

Stocker, A.A. and Simoncelli, E.P. (2005). Sensory adaptation within a Bayesian framework for perception. *Advances in Neural Information Processing Systems (NIPS 2005), Vancouver, Canada* (eds. Y. Weiss, B. Schölkopf, and J. Platt), Vol. 18, pp. 1291 – 1298, MIT Press. [135]

Sung, K. and Poggio, T. (1998). Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 1, pp. 39–51. [95]

Sutton, R.S. and Barto, A.G. (1998). *Reinforcement Learning: An Introduction*. MIT Press, a Bradford Book, Cambridge, MA. [14, 42, 64]

Suzuki, M. and Floreano, D. (2006a). Active vision for neural development and landmark navigation. *50th Anniversary Summit of Artificial Intelligence*, pp. 247–248. [5, 40]

Suzuki, M. and Floreano, D. (2006b). Evolutionary active vision toward three dimensional landmark-navigation. *9th International Conference on the Simulation of Adaptive Behavior (SAB 2006), Rome, Italy* (eds. S. Nolfi, G. Baldassarre, R. Calabretta, J.C.T. Hallam, D. Marocco, J.-A. Meyer, O. Miglino, and D. Parisi), pp. 263–273, Springer, Lecture Notes in Computer Science. [5, 40]

Suzuki, M., van der Blij, J., and Floreano, D. (2006). Omnidirectional active vision for evolutionary car driving. *9th International Conference on Intelligent Autonomous Systems (IAS 2006), Tokyo, Japan* (eds. T. Arai, R. Pfeifer, T.R. Balch, and H. Yokoi), pp. 153–161, IOS Press. [10]

Tarapore, D., Lungarella, M., and Gómez, G. (2006). Quantifying patterns of agent-environment interaction. *Robotics and Autonomous Systems*, Vol. 54, No. 2, pp. 150–158. [63]

te Boekhorst, I.R.J.A., Lungarella, M., and Pfeifer, R. (2003). Dimensionality reduction through sensory-motor coordination. *Artificial Neural Networks and Neural Information Processing (ICANN 2003), Istanbul, Turkey* (eds. O. Kaynak, E. Alpaydin, E. Oja, and L. Xu), pp. 496–503, Springer-Verlag. [63]

Terzopoulos, D. and Rabie, T. F. (1997). Animat vision: active vision in artificial animals. *Videre: Journal of Computer Vision Research*, Vol. 1, No. 1, pp. 2–19. [9, 39]

Terzopoulos, D., Rabie, T., and Grzesczuk, R. (1996). Perception and learning in artificial animals. *5th International Conference on the Synthesis and Simulation of Living Systems (Artificial Life V), Nara, Japan*, pp. 346–353, MIT Press, Cambridge, MA. [39]

Thornton, C. (2005). Principled exploitation of behavioural coupling. *Unpublished MS, Department of Informatics, University of Sussex*. [63]

Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. MIT Press, Cambridge, MA. [64]

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.-E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A., and Mahoney, P. (2006). Stanley: The robot that won the DARPA Grand Challenge. *Journal of Field Robotics*, Vol. 23, No. 9, pp. 661–692. [74]

Torben-Nielsen, B., de Croon, G.C.H.E., and Postma, E.O. (2005). Timing is important: delaying action execution in Plastic Neural Networks. *Belgian-Dutch AI Conference (BNAIC 2005)* (eds. K. Verbeeck, K. Tuyls, A. Nowé, B. Manderick, and B. Kuijpers), pp. 232–238, Royal Flemish Academy of Belgium for Science and Arts (KVAB), Brussels, Belgium. [181]

Torralba, A. (2003). Contextual priming for object detection. *International Journal of Computer Vision*, Vol. 53, No. 2, pp. 169–191. [96, 115, 131, 140]

Torralba, A., Murphy, K. P., and Freeman, W. T. (2004a). Sharing features: efficient boosting procedures for multiclass object detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2004), Washington, DC*, pp. 762–769, IEEE Computer Society. [95]

Torralba, A., Murphy, K., and Freeman, W. T. (2004b). Contextual models for object-detection using boosted random fields. *Advances in Neural Information Processing Systems (NIPS 2004), Vancouver, Canada*, Vol. 17, pp. 1401–1408. [95]

Torralba, A., Oliva, A., Castelhano, M., and Henderson, J. M. (2006). Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. *Psychological Review*, Vol. 113, No. 4, pp. 766–786. [38, 96, 135]

Triesch, J., Ballard, D., Hayhoe, M.M., and Sullivan, B.T. (2003). What you see is what you need. *Journal of Vision*, Vol. 3, No. 1, pp. 86–94. [2]

Tuci, E., Trianni, V., and Dorigo, M. (2004). 'Feeling' the flow of time through sensorimotor co-ordination. *Connection Science*, Vol. 16, No. 4, pp. 301–324. [135]

van Beers, R.J. and Wolpert, D.M. (2002). When feeling is more important than seeing in sensorimotor adaptation. *Current Biology*, Vol. 12, No. 10, pp. 834–837. [135]

van Dartel, M. F., Postma, E. O., van den Herik, H. J., and de Croon, G.C.H.E. (2004). Macroscopic analysis of robot foraging behaviour. *Connection Science*, Vol. 16, No. 3, pp. 169 – 181. [181]

van Dartel, M. F., Sprinkhuizen-Kuyper, I. G., Postma, E. O., and van den Herik, H. J. (2005). Reactive agents and perceptual ambiguity. *Adaptive Behavior*, Vol. 13, No. 3, pp. 227–242. [10, 67, 128, 139]

van der Maaten, L.J.P., Postma, E.O., and van den Herik, H.J. (submitted). Dimensionality Reduction: A Comparative Review. *preprint*. [131]

van Gelder, T.J. (1999). Dynamic approaches to cognition. *The MIT Encyclopedia of Cognitive Sciences* (eds. R. Wilson and F. Keil), pp. 243–245. MIT Press, Cambridge, MA. [135]

van Lankveld, T., de Croon, G.C.H.E., and Tuyls, K. (2007). Static versus plastic controllers in evolutionary robotics. *19th Belgian-Dutch Conference on Artificial Intelligence (BNAIC 2007), Utrecht, the Netherlands* (eds. M. Dastani and E. de Jong), pp. 196–204. [182]

van Zaanen, M. and de Croon, G.C.H.E. (2004). Multi-modal information retrieval using FINT. *ImageCLEF 2004 Workshop, Bath, UK* (eds. C. Peters, P. Clough, J. Gonzalo, G.J.F. Jones, M. Kluck, and B. Magnini), pp. 728–739. [181]

Varela, F.J., Maturana, H.R., and Uribe, R. (1974). Autopoiesis: the organization of living systems, its characterization and a model. *Biosystems*, Vol. 5, No. 4, pp. 187–196. [3]

Viola, P. and Jones, M. J. (2001). Robust real-time object detection. Technical report, Cambridge Research Laboratory. [42, 47, 94, 95, 104, 107, 109, 120, 123, 130]

Vogel, J. and Murphy, K. (2007). A non-myopic approach to visual search. *4th Canadian Conference on Computer and Robot Vision*, pp. 227–234, CRV. [97]

Wann, J.P. and Wilkie, R.M. (2004). How do we control high speed steering? *Chapter 18 in 'Optic flow and beyond'* (eds. L.M. Vaina, S.A. Beardsley, and S.K. Rushton), pp. 401 – 419. [70, 92]

Weber, M. (2000). *Unsupervised learning of models for object recognition*. Ph.D. thesis, California Institute of Technology, Pasadena, CA. [95]

Weber, M., Welling, M., and Perona, P. (2000a). Towards automatic discovery of object categories. *IEEE conference on Computer Vision and Pattern Recognition (CVPR 2000), Hilton Head Island, SC*, pp. 101–109, IEEE Computer Society. [95]

Weber, M., Welling, M., and Perona, P. (2000b). Unsupervised learning of models for recognition. *6th European Conference on Computer Vision (ECCV 2000), Dublin, Ireland*, Vol. 1, pp. 18–32, Springer-Verlag. [95]

Weiss, Y., Simoncelli, E.P., and Adelson, E.H. (2002). Motion illusions as optimal percepts. *Nature Neuroscience*, Vol. 5, pp. 598 – 604. [135]

Wilkie, R.M. and Wann, J.P. (2003). Controlling steering and judging heading: retinal flow, visual direction, and extraretinal information. *Journal of Experimental Psychology*, Vol. 29, No. 2, pp. 363–378. [70, 92]

Wolf, L. and Bileschi, S. (2006). A critical view of context. *International Journal of Computer Vision*, Vol. 69, No. 2, pp. 251–261. [95, 115, 119]

Wolpert, D.M. and Ghahramani, Z. (2000). Computational principles of movement neuroscience. *Nature neuroscience*, Vol. 3, pp. 1212–1217. [135]

Xu, N., Ahuja, N., and Bansal, R. (2007). Object segmentation using graph cuts based active contours. *Computer Vision and Image Understanding*, Vol. 107, No. 3, pp. 210–224. [131]

Yarbus, A. L. (1967). *Eye movements and vision*. Plenum, New York, NY. [2]

Young, S. S., Scott, P. D., and Bandera, C. (1998). Foveal automatic target recognition using a multiresolution neural network. *IEEE Transactions on Image Processing*, Vol. 7, No. 8, pp. 1122–1135. [39, 97, 131]

Yuille, A.L. (1991). Deformable templates for face recognition. *Journal of Cognitive Neuroscience*, Vol. 3, No. 1, pp. 59 – 70. [131, 133]

Zhao, W.Y. and Chellappa, R. (2002). Image-based Face Recognition: Issues and Methods. *Image Recognition and Classification* (eds. B. Javidi and M. Dekker), pp. 375–402. [131]

Zhoua, J., Lua, X., Zhang, D., and Wua, C. (2002). Orientation analysis for rotated human face detection. *Image and Vision Computing*, Vol. 20, No. 4, pp. 257–264. [133]

Zuo, F. and de With, P.H.N. (2004). Fast facial feature extraction using a deformable shape model with Haar-wavelet based local texture attributes. *International Conference on Image Processing (ICIP 2004), Singapore*, Vol. 3, pp. 1425–1428. [131, 133]

Appendix

A

Derivations for Chapter 2

This appendix is based on the following publication[1]:

1. de Croon, Sprinkhuizen-Kuyper, and Postma, 2006c, *Comparing Active Vision Models.* MICC-IKAT Technical Report 06-02.

In this appendix, we derive two of the formulas used in Chapter 2. In Section A.1 we derive the belief state update that is used by all active vision models in Chapter 2. In Section A.2 we derive the formula for action selection in the model MI. Both derivations are based on the work in Denzler and Brown (2000) and in Denzler and Brown (2002).

A.1 Belief State Update

Here we derive the recursive update for the belief state, as used in Denzler and Brown (2002). The belief state is the posterior class probability distribution, which we can rewrite using Bayes' rule as follows.

$$p(C \mid \mathbf{o}_i, \mathbf{a}_i) = \frac{p(o_i \mid C, \mathbf{o}_{i-1}, \mathbf{a}_i) p(C \mid \mathbf{o}_{i-1}, \mathbf{a}_i)}{p(o_i \mid \mathbf{o}_{i-1}, \mathbf{a}_i)} \tag{A.1}$$

In Denzler and Brown (2002) the classification task concerns the classification of different 3-D objects. Each action of the active vision model corresponds to an angle from which an object can be viewed. Since in their experimental setup any angle can be reached at any time step, it is assumed that an observation is only determined by the class and the action, and not by the past observations and actions:

$$p(o_i \mid C, \mathbf{o}_{i-1}, \mathbf{a}_i) = p(o_i \mid C, a_i). \tag{A.2}$$

[1]The author would like to thank his co-authors for their permission to use parts of the publication in this appendix.

This assumption, referred to as the Markov assumption, can be used to rewrite the formula of the posterior class probability distribution as follows.

$$p(C \mid \mathbf{o}_i, \mathbf{a}_i) = \frac{p(o_i \mid C, a_i)p(C \mid \mathbf{o}_{i-1}, \mathbf{a}_i)}{p(o_i \mid \mathbf{o}_{i-1}, \mathbf{a}_i)} \tag{A.3}$$

The formula can further be simplified by using the fact that $p(C \mid \mathbf{o}_{i-1}, \mathbf{a}_i) = p(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})$, since action a_i does not contain any information on the class if o_i is unknown. As a consequence, the denominator $p(o_i \mid \mathbf{o}_{i-1}, \mathbf{a}_i)$ can be rewritten as follows.

$$p(o_i \mid \mathbf{o}_{i-1}, \mathbf{a}_i) = \tag{A.4}$$

$$\sum_{c \in C} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_i)p(o_i \mid c, \mathbf{o}_{i-1}, \mathbf{a}_i) =$$

$$\sum_{c \in C} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})p(o_i \mid c, \mathbf{o}_{i-1}, \mathbf{a}_i) =$$

$$\sum_{c \in C} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})p(o_i \mid c, a_i) \tag{A.5}$$

The rewritten formula of the posterior class probability distribution becomes:

$$p(C \mid \mathbf{o}_i, \mathbf{a}_i) = \frac{p(o_i \mid C, a_i)p(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})}{\sum_{c \in C} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})p(o_i \mid c, a_i)}. \tag{A.6}$$

As explained in Chapter 2, this formula can be used as a recursive belief state update.

A.2 Action Selection MI

In the following, we derive the formula for the action selection of MI (based on Denzler and Brown, 2000; Denzler and Brown, 2002). MI selects an action a to maximise the mutual information (see, e.g., Cover and Thomas, 1991) between the classes and observations, given the past observations and actions:

$$I(C; O \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) = \tag{A.7}$$

$$H(C \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) - H(C \mid O, \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) =$$

$$H(O \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) - H(O \mid C, \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) =$$

$$H(O \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) - \sum_{c \in C} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) H(O \mid c, \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)$$

$$\begin{array}{c} = -\sum_{o \in O} p(o \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) \log(p(o \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)) + \\ \sum_{c \in C} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) \sum_{o \in O} p(o \mid c, \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) \log(p(o \mid c, \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)) \end{array}$$

$$\begin{array}{c} = -\sum_{o \in O} p(o \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) \log\left(p(o \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)\right) + \\ \sum_{c \in C} \sum_{o \in O} p(o \mid c, \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) \log(p(o \mid c, \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)) \end{array}$$

$$\begin{array}{c} = -\sum_{c \in C} \sum_{o \in O} p(o, c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) \log(p(o \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)) + \\ \sum_{c \in C} \sum_{o \in O} p(o, c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) \log(p(o \mid c, \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)) \end{array}$$

$$= \sum_{c \in C} \sum_{o \in O} p(o, c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) \log\left(\frac{p(o \mid c, \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)}{p(o \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)}\right)$$

$$= \sum_{c \in C} \sum_{o \in O} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) p(o \mid c, \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) \log\left(\frac{p(o \mid c, \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)}{p(o \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a)}\right). \quad \text{(A.8)}$$

Note that $p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) = p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1})$, since the probabilities of C are independent of the action that has not yet been executed. In addition, $p(o \mid c, \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) = p(o \mid c, a)$, according to the assumption that observations only depend on the object class and the viewing angle (see Section 2.2). As a result, the formula of the mutual information is simplified to:

$$I(C; O \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}, a) = \quad \text{(A.9)}$$

$$\sum_{c \in C} \sum_{o \in O} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}) p(o \mid c, a) \log\left(\frac{p(o \mid c, a)}{\sum_{c \in C} p(c \mid \mathbf{o}_{i-1}, \mathbf{a}_{i-1}) p(o \mid c, a)}\right). \quad \text{(A.10)}$$

MI selects the action a that maximises this measure.

Appendix

B

Analysis Car-detection Task

In this appendix, we present the analysis of the instance of ACT-DETECT that is adapted to the car-detection task. We analyse this instance as in Section 6.6: we start with an analysis of the evolved visual features, and then study its mapping from inputs to gaze shifts in the image.

B.1 Visual Features

Figure B.1 shows the twenty evolved features for the car-detection task and Figure B.2 shows the responses of these features on all locations of an example image. The features are somewhat more difficult to analyse than those of the face-detection task. The reason for this is the higher variation in backgrounds, object sizes, and object viewpoints. However, after studying many figures such as Figure B.2, some regularities became evident. As in the face-detection task, there are features that respond to the object's context, and features that respond to the object itself.

Features 7, 8, and 17 are involved in detecting the border of the road and the bottom of larger cars. In Figure B.2 this is clearest for feature 17, since it only has a high response (indicated with high intensity) just above the curb, and on the car. In the figure the high responses of feature 7 and 8 are not only limited to the curb, but also to other locations in the image. Furthermore, the small centre surround feature 20 seems to play a role in detecting contextual clues. It has an average response in areas with little texture (such as the sky, or the building in the image) and high and low responses in highly textured areas.

Features 1, 4, 5, 6, 9, 12, 16, and 19 respond to car-features. Note that some of these features are very similar in shape and type, but are complementary in their position in the gaze window. For example, features 1 and 6 detect horizontal contrasts at different distances of the gaze location (indicated with the white cross). They often respond to the border between a car and the background. Features 4 and 19 are similar to features 1 and 6, although they seem to detect the contrasts at a slightly different resolution, and at different places. All four of these features (1, 4, 6, and 19) have low responses close to the car shown in the original image

Figure B.1: The twenty evolved features for the car-detection task, shown within the gaze window (white box) of the coarse-scale model. The white cross indicates the centre of the gaze window.

of Figure B.2. Feature 16 is a very interesting feature: it responds very strongly to the tires of the larger cars. Figure B.2 shows that the response of the feature is very high or low to the top left of the car's tires. The feature also responds to the side of smaller cars.

As in Section 6.6, we measure the information contained in clusters of visual inputs on the location of the closest object. We gathered 30 samples for 132 training images by uniformly sampling in the image. We clustered these inputs with k-means clustering for $k = \langle 2, 3, 4, \ldots, 15 \rangle$. Figure B.3 shows the relation between the number of clusters and mutual information with the horizontal displacement ΔX (solid line) and vertical displacement ΔY (dashed line) to the closest object. Note that the amount of information is smaller than for the face-detection task (Section 6.6). We might augment this information by expanding the set of features that can be selected by the evolutionary algorithm.

Figure B.4 shows the spatial distributions for $k = 6$ clusters (the order of the clusters is irrelevant). The closest object is shown in the middle of every inset with a white cross, and white regions indicate regions where the input cluster has a high probability of occurring (black regions indicate the opposite). The figure shows that clusters 1 and 2 are mostly situated above the object, while cluster 4 occurs below the object. Clusters 3, 5, and 6 have a light bias towards locations above the closest object.

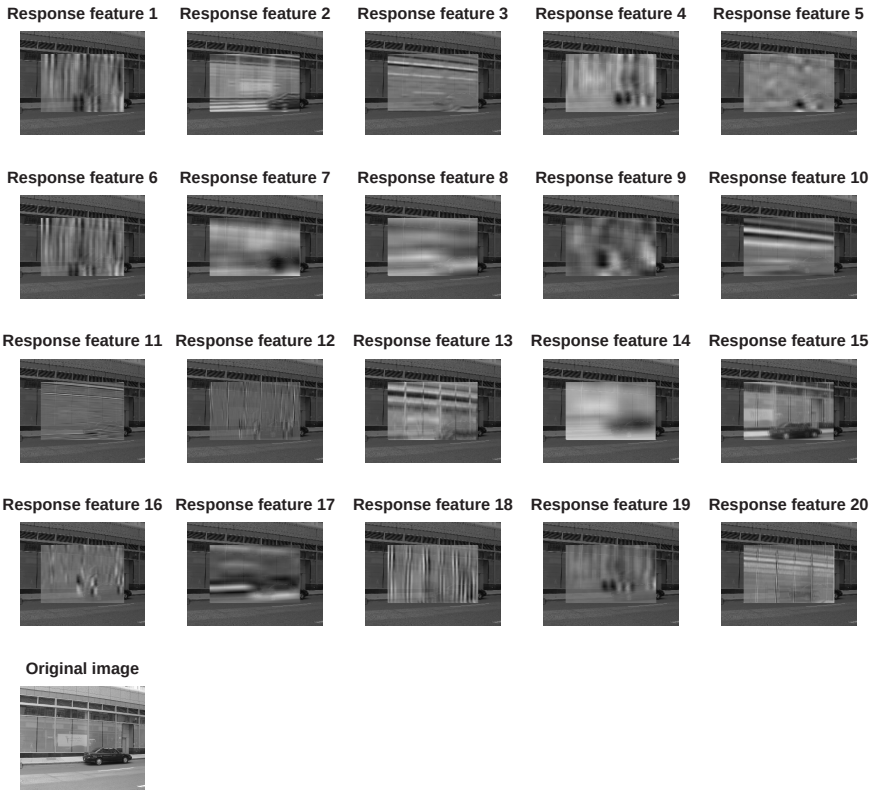


Figure B.2: Responses of the twenty features shown in Figure B.1 in different parts of the image. The evolved features have been extracted at all possible gaze locations in an image (labelled 'original image'). Per feature, we show a darkened version of the image as the background, and the feature responses on the foreground. The feature responses are scaled to [0,1], and represented with gray-values. White implies a high response, black a low response. We show the response at the scanning location, i.e., the centre of the gaze window when the feature was extracted.

B.2 Gaze Shifts

We first show that ACT-DETECT makes different gaze shifts for inputs that occur at different places relative from the closest object. Then, we measure the relation between the information in the features and the influence they have on ACT-DETECT's actions. Finally, we analyse the effects of the prior object distribution of the car-detection image set on ACT-DETECT's behaviour.

In Figure B.4 we show the gaze shifts made by the coarse-scale model, when we give the cluster centroids as inputs to its controller. It reacts to the spatial distribution of the clusters as expected: by mapping the input to a gaze shift that approaches the closest object location. This is best illustrated by the shift for input

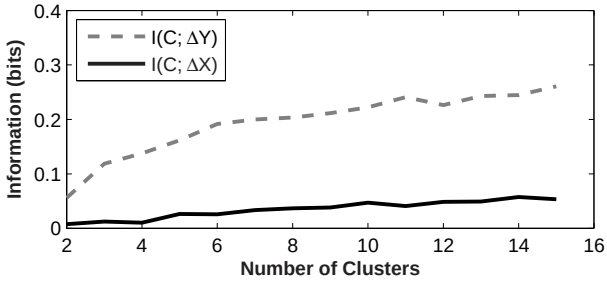


Figure B.3: The relation between the number of clusters and the mutual information of the clustering with the horizontal distance (ΔX, solid line) and vertical distance (ΔY, dashed line) to the closest object. The mutual information is expressed in bits.

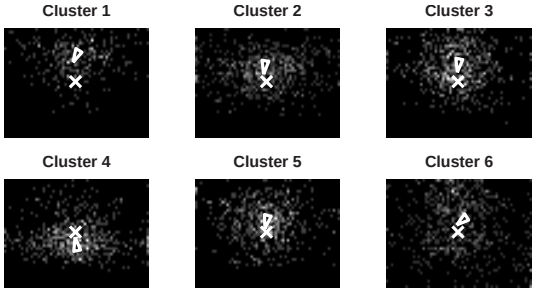


Figure B.4: Spatial distribution of six visual input clusters relative to the closest object. High occurrence is represented with high intensity, low occurrence with low intensity. The location of the closest object is shown with a white cross. The arrows originate at the median of all locations in the cluster, and represent the actions taken by ACT-DETECT if it receives the cluster centroid as visual input.

cluster 4, since it is the only cluster below the object. As a reaction to the cluster centroid, ACT-DETECT shifts its gaze upwards. Note that the model does not follow a greedy action strategy. In that case, one would expect the model to make larger gaze shifts, e.g., for cluster 1. In Figure B.5 we see the gaze shifts of the coarse-scale model at all points of a 10×10 grid (left) and the gaze shifts of the fine-scale model at all points of a 20×20 grid (right).

The coarse-scale model uses the information in the individual features to determine its actions. However, for the car-detection task this is only clear for the vertical displacement. Table B.2 shows the mutual information of each feature and the horizontal and vertical displacement to the closest target, $I(F_i; \Delta X)$ and $I(F_i; \Delta Y)$. In addition, it shows the information on the horizontal and vertical part of ACT-DETECT's gaze shift, $I(F_i; A_{\Delta X})$ and $I(F_i; A_{\Delta Y})$. Features with more information on the vertical displacement contain more information on the vertical part of the ACT-DETECT's gaze shift. The features with most information on the ACT-DETECT's horizontal shifts are features 5, 7, 9, and 19. For features 5, 9, and 19 this is related

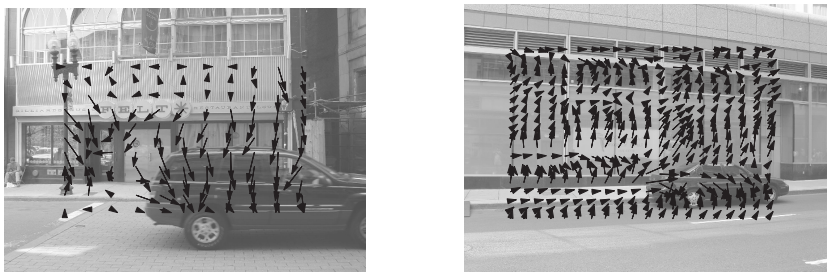


Figure B.5: **Left:** Actions taken by the coarse-scale model on a 10×10 grid in the image. **Right:** Actions taken by the fine-scale model on a 20×20 grid in the image.

to the fact that they respond to the borders of the car. For feature 7 it may be a behavioural choice: when detecting the curb the model moves horizontally until it detects car-related features.

Table B.1: Mutual information between the evolved features F_i, $i \in \{1, 2, 3, ..., 20\}$ and the horizontal displacement to the closest object ΔX, the vertical displacement ΔY, the horizontal part of the gaze shift $A_{\Delta X}$, and the vertical part $A_{\Delta Y}$.

	$I(F_i; \Delta X)$	$I(F_i; \Delta Y)$	$I(F_i; A_{\Delta X})$	$I(F_i; A_{\Delta Y})$
$i = 1$	0.03	0.06	0.04	0.02
$i = 2$	0.03	0.06	0.02	0.05
$i = 3$	0.02	0.05	0.05	0.07
$i = 4$	0.03	0.07	0.04	0.04
$i = 5$	0.02	0.04	0.09	0.03
$i = 6$	0.02	0.05	0.03	0.01
$i = 7$	0.03	0.13	0.11	0.33
$i = 8$	0.03	0.05	0.02	0.03
$i = 9$	0.03	0.04	0.13	0.04
$i = 10$	0.02	0.09	0.02	0.14
$i = 11$	0.02	0.04	0.02	0.03
$i = 12$	0.02	0.05	0.01	0.02
$i = 13$	0.02	0.08	0.03	0.04
$i = 14$	0.04	0.12	0.03	0.48
$i = 15$	0.03	0.07	0.02	0.17
$i = 16$	0.02	0.07	0.06	0.06
$i = 17$	0.02	0.09	0.02	0.04
$i = 18$	0.03	0.03	0.02	0.02
$i = 19$	0.03	0.06	0.32	0.06
$i = 20$	0.03	0.06	0.03	0.06

The prior distribution of the CBCL StreetScenes database is much less strong than that of the FGNET data set. The crosses in Figure B.6 represent the car locations in the data set (centre of the cars). Surprisingly, this weaker prior dis-

tribution results in a slightly larger relative prior with uniform sampling than for the FGNET data set: the median $(\Delta x, \Delta y) = (5.0, 48.0)$. If we use the circular sampling explained in Subsection 6.6.2, the median replacement is reduced to $(\Delta x, \Delta y) = (1.0, 15.0)$.

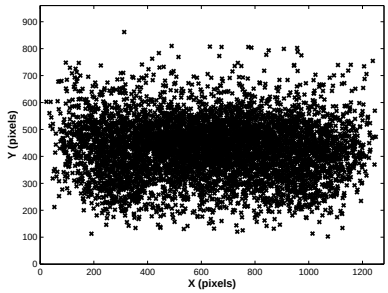


Figure B.6: Prior distribution of car locations in the CBCL StreetScenes data set. Crosses show the locations of the labelled cars in the data set in image coordinates.

Summary

Actions are essential to visual perception. Humans and animals rely heavily on movements of their eyes, head, and body to perceive the environment in which they live. It has been shown that computer vision models can also use visual actions to improve their performance on visual tasks. However, the problem with developing such *active vision models* is that we do not know how they should determine their actions. A typical approach to this problem is to make assumptions on the vision process. For example, a main assumption is that visual actions have as goal to gather information on a predefined part of the world. Such an assumption facilitates designing an action strategy for an active vision model.

In the thesis, we employ a different approach to the problem of determining an action strategy: we *adapt* active vision models to their task. Currently, there are three main obstacles on the way towards a successful application of adaptive active vision models in computer vision. First, it is uncertain whether such models are able to cope with visually challenging tasks. Second, the manner in which the models handle visual tasks is poorly understood. Third, active vision is always associated with servo-motor systems. It is not clear whether adaptive active vision models can also improve computer vision involving static image sets. To overcome the three obstacles, we formulate the following problem statement.

> **Problem statement**: *How do adaptive active vision models handle challenging visual tasks?*

A visual task is challenging if humans are currently still better at it than state-of-the-art computer vision algorithms. In most of the challenging tasks studied, we use a model that only processes local image samples. In our opinion, such a model is most promising for the field of computer vision; it is computationally efficient and can be applied to static images. Since the model has to direct its *gaze* to elements of interest in the visual scene, we refer to it as a *gaze control model*. We attempt to gain insight into the problem statement by answering four research questions (RQs), each in a separate chapter.

RQ 1: How does the adaptive active vision approach relate to other approaches of active vision?

RQ 2: How does a memoryless adaptive gaze control model handle an image classification task?

RQ 3: How does an adaptive gaze control model use its gaze shifts in a control task?

RQ 4: Can an adaptive gaze control model perform on a par with state-of-the-art computer vision models on the task of object detection?

We start our research in Chapter 2 by studying RQ 1: *How does the adaptive active vision approach relate to other approaches of active vision?* In the literature, there is one other main approach to active vision, which we name the *probabilistic approach*. This approach assumes that active vision is an iterative process of state estimation and action selection. The central goal of the actions of probabilistic models is to reduce uncertainty on a predetermined part of the world state. In contrast to the probabilistic approach, the *adaptive approach* makes no assumptions on the active vision process. Studies of this approach adapt a predetermined structure (such as a neural network) to optimise performance on the task. We first describe models from both approaches using the same notation. Then, we compare the active vision models with each other, by applying them to a task of 3-D object classification. The results show that the adaptive model performs as well as the probabilistic models, despite its lack of a formal action-selection framework.

In Chapter 3 we introduce a framework for the gaze control models that we use to answer the remaining three research questions (RQ 2, 3, and 4). We evaluate existing gaze control models on the basis of three requirements: local processing, task-dependency, and making no assumptions on what gaze strategy to follow. Our gaze control framework is inspired by the group of gaze control models that fulfil all three requirements. The main difference between our model and this group lies in the application of the model to visually challenging tasks. As a consequence, we use visual features that are different from the ones used by previous models.

Subsequently, in Chapter 4, we focus on RQ 2: *How does a memoryless adaptive gaze control model handle an image classification task?* To answer this question, we apply our gaze control model to the task of gender recognition in static natural images and compare it with a passive gaze control model. The active model outperforms the passive model on the task. Our analysis shows that the gaze control model shifts its gaze to image areas that are better for classification; the model optimises the information that its observations contain on the image class. By using its gaze location as a form of external memory, it can exploit multiple observations in spite of the memoryless nature of the controller.

Then, in Chapter 5 we investigate RQ 3: *How does an adaptive gaze control model use its gaze shifts in a control task?* A control task is different from a classification task, since the goal of the task is successful behaviour, not purely the gathering of information. Since it is well-known that humans' gaze shifts also serve other purposes than information gathering, we find it interesting to investigate whether an adaptive model uses its gaze shifts in a control task solely for information gathering. In particular, we apply an adaptive model to the task of car-driving in a simulator. The analysis shows that the gaze shifts of an evolved agent serve the following three functions: (1) to find relevant visual features in the beginning of a run, (2) to keep relevant visual features in sight, and (3) to avoid disruptive visual inputs. Where the first two functions concern the gathering of information from the

environment, the third function does not. By avoiding disrupting visual inputs, the model obviates the need for complex internal processing of such inputs.

In Chapter 6, we turn to RQ 4: *Can an adaptive gaze control model perform on a par with state-of-the-art computer vision models on the task of object detection?* Existing object-detection methods usually perform an exhaustive scan of an image by evaluating local samples at all positions in the image either as being an object or being part of the background. We introduce a gaze control model that uses the information that local samples may contain on probable object locations. The goal of employing a gaze control model for object detection is to enhance computational efficiency, while retaining as good a detection performance as possible. We perform experiments on a face-detection task and a car-detection task, which show that the gaze control model can perform on a par with state-of-the-art object-detection methods. Our analysis demonstrates that the gaze control model is computationally much more efficient than the other methods, but that the exploitation of contextual information leads to a reduced generality.

Chapter 7 contains the general discussion of the findings in the thesis. First, we discuss all findings in the thesis regarding the differences between probabilistic and adaptive active vision. Then, we estimate to what extent our results can be generalised to other contexts. Subsequently, we address the difference between passive vision and active vision in static images. We also discuss in what cases adaptive active vision models can contribute to the understanding of natural vision.

In Chapter 8, we answer the problem statement on the basis of the results. Adaptive active vision models handle challenging visual tasks by exploiting both their internal processing and their feedback loop with their environment. This feedback loop allows the models to facilitate the execution of their task. They can use the feedback loop to maximise task-specific information in their observations, by using the environment as an external memory. They can also use the feedback loop to avoid disruptive visual inputs. This obviates the need for complex internal processing of such inputs.

Finally, we may conclude that adaptive active vision is a promising approach for discovering the role of actions in vision. The approach has its restrictions and problems, but making fewer assumptions allows the models to find strategies that can both contribute to (1) a better understanding of the active vision process and (2) the improvement of computer vision techniques.

Samenvatting

Acties zijn essentieel voor visuele waarneming. Mensen en dieren gebruiken hun ogen, hoofd, en lichaam om de wereld om hen heen te zien. Computermodellen kunnen ook gebruik maken van acties. Het is aangetoond dat het gebruik van acties bepaalde visuele taken makkelijker kan maken. Het probleem met zulke *actieve waarnemingsmodellen* is dat we voor veel visuele taken niet weten hoe ze hun acties zouden moeten bepalen. Een standaard aanpak van dit probleem is om een aantal aannames te maken over het visuele proces. Een hoofdaanname in veel actieve waarnemingsmodellen is bijvoorbeeld dat een visuele actie (zoals een oogbeweging) als doel heeft om informatie te vergaren over de omgeving. Zo'n aanname vergemakkelijkt het vinden van een actiestrategie voor een actief waarnemingsmodel.

In het proefschrift gebruiken we een andere aanpak om een actiestrategie te vinden. We bepalen niet op voorhand wat de modellen moeten doen, maar passen ze aan specifieke taken aan. We noemen zulke modellen *adaptieve actieve waarnemingsmodellen*. Op het moment zijn er drie obstakels die verhinderen dat zulke modellen succesvol gebruikt kunnen worden op het gebied van 'computer vision'. Ten eerste is het onzeker of adaptieve actieve waarnemingsmodellen succesvol kunnen worden toegepast op uitdagende visuele taken. Ten tweede is het onduidelijk *hoe* een aangepast model zijn acties bepaalt. Ten derde wordt actieve waarneming altijd geassocieerd met fysieke camera's die kunnen bewegen. Het is niet duidelijk of actieve waarnemingsmodellen ook iets zouden kunnen betekenen voor visuele taken in statische plaatjes. Om deze drie obstakels weg te nemen, formuleren we de volgende probleemstelling.

> **Probleemstelling**: *Hoe pakken adaptieve actieve waarnemingsmodellen uitdagende visuele taken aan?*

We noemen een visuele taak uitdagend als mensen nog steeds beter zijn in het uitvoeren van de taak dan 'state-of-the-art' computer vision modellen. In het merendeel van de bestudeerde visuele taken gebruiken we een actief waarnemingsmodel dat alleen lokale delen van een plaatje gebruikt om zijn taak te vervullen. De reden hiervoor is dat zo'n model veelbelovend is voor computer vision: het is computationeel efficiënt en kan gebruikt worden in statische plaatjes. Het model maakt virtuele oogbewegingen om de relevante delen van het plaatje op te zoeken. We refereren naar dit model als het *oogbewegingsmodel*. We proberen inzicht te verkrijgen in de probleemstelling door vier onderzoeksvragen te beantwoorden, die elk in

een apart hoofdstuk behandeld worden. We duiden de vragen aan met de afkorting RQ, van het Engelse 'Research Question'.

RQ 1: Hoe verhoudt de adaptieve aanpak van actieve waarneming zich tot andere aanpakken?

RQ 2: Hoe pakt een geheugenloos adaptief oogbewegingsmodel een classificatietaak in plaatjes aan?

RQ 3: Hoe gebruikt een adaptief oogbewegingsmodel zijn oogbewegingen bij het uitvoeren van een controletaak?

RQ 4: Kan een adaptief oogbewegingsmodel even goed zijn als state-of-the-art computer vision modellen in het detecteren van objecten in plaatjes?

We beginnen ons onderzoek in hoofdstuk 2 met het bestuderen van RQ 1: *Hoe verhoudt de adaptieve aanpak van actieve waarneming zich tot andere aanpakken?* Ons literatuuronderzoek leidt tot het inzicht dat er twee gescheiden aanpakken van active vision zijn. De *probabilistische aanpak* neemt aan dat actieve waarneming een iteratief proces is waarin telkens eerst de toestand van de wereld geschat wordt, en dan een actie geselecteerd wordt. Het centrale doel van de acties van probabilistische modellen is het verkrijgen van meer zekerheid over de toestand van de wereld. De *adaptieve aanpak* hebben we boven al kort besproken. Deze aanpak maakt geen aannames over hoe het model zijn acties moet bepalen. In plaats van het ontwerpen van een actiestrategie, wordt een voorbepaald model aan een taak aangepast met als doel het optimaliseren van de prestatie van het model. Eerst beschrijven we de modellen van beide aanpakken in een gezamenlijke notatie. Daarna vergelijken we de modellen met elkaar op een classificatietaak van drie-dimensionale objecten. De resultaten tonen aan dat het adaptieve model even goed presteert als de probabilistische modellen, ondanks zijn gebrek aan een formele actiestrategie.

In hoofdstuk 3 introduceren we het oogbewegingsmodel dat we zullen gebruiken om de overige drie onderzoeksvragen te beantwoorden (RQ 2, 3, en 4). We evalueren andere bestaande oogbewegingsmodellen in de literatuur met betrekking tot drie voorwaarden voor ons model: het uitsluitend gebruik van lokale delen van een plaatje, het hebben van een taakafhankelijke actiestrategie, en het niet maken van aannames over de actiestrategie. Ons oogbewegingsmodel is geïnspireerd door de groep van modellen die aan de voorwaarden voldoen. Het voornaamste verschil van ons model met deze groep ligt in de toepassing van het model op uitdagende visuele taken. Daarom gebruiken we andere visuele kenmerken dan voorgaande modellen.

Vervolgens richten we ons in hoofdstuk 4 op RQ 2: *Hoe pakt een geheugenloos adaptief oogbewegingsmodel een classificatietaak in plaatjes aan?* Om deze vraag te beantwoorden, passen we ons oogbewegingsmodel toe op een taak van genderherkenning (man of vrouw) in statische plaatjes. We vergelijken het actieve oogbewegingsmodel met een passief oogbewegingsmodel om inzicht te krijgen in de meerwaarde van de acties. Het actieve model presteert beter dan het passieve model. Onze analyse laat zien dat het actieve model zijn observaties gebruikt om naar delen van het plaatje te gaan die meer geschikt zijn voor classificatie; het model

maximaliseert de informatie die een enkele observatie bevat over de klasse van het plaatje. Hierbij gebruikt het model zijn fixatielocatie als een vorm van extern geheugen, zodat het ondanks zijn gebrek aan intern geheugen toch meerdere observaties kan gebruiken voor classificatie.

Daarna, in hoofdstuk 5 onderzoeken we RQ 3: *Hoe gebruikt een adaptief oogbewegingsmodel zijn oogbewegingen bij het uitvoeren van een controletaak?* In een controletaak is het doel succesvol gedrag, en niet het vergaren van informatie op zich. Aangezien het bekend is dat oogbewegingen voor mensen ook andere doelen kan hebben dan het vergaren van informatie, is het interessant te onderzoeken hoe een adaptief model zijn oogbewegingen gebruikt voor een controletaak. In onze experimenten passen we een adaptief model toe op een taak van autorijden in een simulator. De analyse van het model toont aan dat de oogbewegingen de volgende drie doelen dienen: (1) het vinden van relevante visuele kenmerken, (2) het in de gaten houden van deze kenmerken, en (3) het ontwijken van verstorende waarnemingen. De eerste twee doelen kunnen geïnterpreteerd worden als het vergaren van informatie, maar het derde doel niet. Door verstorende waarnemingen te ontwijken, voorkomt het model dat het een complex intern mechanisme moet ontwikkelen om met die waarnemingen om te gaan.

In hoofdstuk 6, beantwoorden we RQ 4: *Kan een adaptief oogbewegingsmodel even goed presteren als state-of-the-art computer vision modellen op de taak van objectdetectie in plaatjes?* Het overgrote deel van de bestaande methodes van objectdetectie zoeken een plaatje geheel af om alle objecten te vinden. Op alle plaatsen in een plaatje wordt een lokaal monster van het plaatje genomen dat geëvalueerd wordt als (deel van) een object, of als deel van de achtergrond. In beide gevallen wordt de zoektocht na de evaluatie van het monster voortgezet zonder de informatie in het monster te gebruiken. Wij introduceren een oogbewegingsmodel dat de informatie in de lokale monsters gebruikt om de zoektocht naar interessante objecten te stroomlijnen. Het monster kan namelijk informatie bevatten over waar objecten zich waarschijnlijk in het plaatje bevinden. Het doel van het model is om computationeel efficiënter te zijn dan bestaande methoden, terwijl de detectieprestaties zo goed mogelijk behouden blijven. We voeren experimenten uit op een gezichtsdetectietaak en een autodetectietaak. De resultaten tonen aan dat ons oogbewegingsmodel even goed kan presteren als bestaande modellen, terwijl het computationeel efficiënter is. Deze efficiëntie heeft echter een prijs: omdat ons model steunt op de visuele context van een object, is de oplossing die onze methode biedt voor objectdetectie minder algemeen dan standaardoplossingen.

In de algemene discussie in hoofdstuk 7 bespreken we de bevindingen in het proefschrift. Eerst bediscussiëren we alle vindingen in het proefschrift aangaande de probabilistische en adaptieve aanpak van active vision. Daarna bespreken we in hoeverre de verkregen inzichten gegeneraliseerd kunnen worden naar modellen en taken die niet specifiek bestudeerd zijn in dit proefschrift. Dan adresseren we de verschillen tussen passieve en actieve computer vision modellen in statische plaatjes. Uiteindelijk geven we ook aan in welke gevallen adaptieve actieve waarnemingsmodellen meer inzicht kunnen verschaffen in de waarneming door dieren of mensen.

In hoofdstuk 8 geven we met behulp van de resultaten een antwoord op de

probleemstelling. Adaptieve actieve waarnemingsmodellen pakken uitdagende visuele taken aan door middel van hun interne mechanismen en de externe 'feedback' die bestaat tussen de acties en observaties. De modellen gebruiken deze feedback om de uitvoering van hun taak te vergemakkelijken. Zo kunnen de modellen de taakspecifieke informatie in de waarnemingen maximaliseren, door de omgeving als extern geheugen te gebruiken. Ook kunnen de modellen de feedback gebruiken om verstorende visuele waarnemingen te ontwijken. Op die manier hoeft het model geen complex intern mechanisme te hebben om die waarnemingen te verwerken.

We concluderen dat adaptieve actieve waarneming een veelbelovende aanpak is voor het ontdekken van de rol van acties in visuele waarneming. Het heeft zijn restricties en problemen, maar het maken van minder aannamen kan leiden tot actiestrategieën die zowel bijdragen aan een beter begrip van actieve visuele processen als aan het verbeteren van computer vision technieken.

Curriculum Vitae

Guido Cornelis Henricus Eugène de Croon was born in Deurne, the Netherlands, on the 17th of June 1980. He attended secondary school at the Cobbenhage College in Tilburg from 1992 to 1998 and obtained the diploma 'VWO'. He then started his studies in 'Knowledge Engineering' at Maastricht University. After taking on many extracurricular activities, he became inspired by evolutionary robotics. He obtained his M.Sc. degree with a major in Artificial Intelligence on the basis of research carried out at the CNR in Rome, in Stefano Nolfi's laboratory. There he studied how the capacities of evolved robots are related to the mathematical properties of their recurrent neural network controllers. He graduated in December 2003.

Subsequently, he started to work as a Ph.D. student at the Maastricht ICT Competence Centre (MICC) of Maastricht University. Funded by the NWO project VINDIT (grant number 634.000.018), he investigated *adaptive active vision*. The goal of his research was to apply methods from evolutionary robotics to computer vision problems, both to gain further insight into the role of actions in vision, and to improve computer vision algorithms. During his Ph.D. period he also performed research for six months at the Ecole Polytechnique Fédérale de Lausanne (EPFL), at Dario Floreano's laboratory.

Besides performing research, he was involved in lecturing (mainly in the courses Informatics 1 and Situated Agents), organising the Perceptual Cognition Workshop (2006), and in coordinating a SIKS / MICC-IKAT colloquium series.

Publications

The investigations performed during my Ph.D. research resulted in the following publications.

1. M.F. van Dartel, E.O. Postma, H.J. van den Herik, and G.C.H.E. de Croon (2004). Macroscopic analysis of robot foraging behaviour. *Connection Science*, Vol. 16, No. 3, pp. 169 – 181.

2. M. van Zaanen and G.C.H.E. de Croon (2004). Multi-modal information retrieval using FINT. *ImageCLEF Workshop, Bath, UK* (eds. C. Peters, P. Clough, J. Gonzalo, G.J.F. Jones, M. Kluck, and B. Magnini), pp. 728 – 739.

3. G.C.H.E. de Croon, M.F. van Dartel, and E.O. Postma (2005). Evolutionary learning outperforms reinforcement learning on non-Markovian tasks. *The Learning and Memory Workshop of the European Conference on Artificial Life (ECAL 2005), Canterbury, UK*, workshop articles on CD.

4. G.C.H.E. de Croon, E.O. Postma, and H.J. van den Herik (2005). A situated model of active vision. *Belgian-Dutch AI Conference (BNAIC 2005), Brussels, Belgium* (eds. K. Verbeeck, K. Tuyls, A. Nowé, B. Manderick, and B. Kuijpers), pp. 74 – 80.

5. B. Torben-Nielsen, G.C.H.E. de Croon, and E.O. Postma (2005). Timing is important: delaying action execution in plastic neural networks. *Belgian-Dutch AI Conference (BNAIC 2005), Brussels, Belgium* (eds. K. Verbeeck, K. Tuyls, A. Nowé, B. Manderick, and B. Kuijpers), pp. 232 – 238.

6. G.C.H.E. de Croon, E.O. Postma, and H.J. van den Herik (2005). Sensory-motor coordination in gaze control. *Applications of Evolutionary Computing (EvoWorkshops 2005), Lausanne, Switzerland* (ed. F. Rothlauf), pp. 334-344, Springer-Verlag, (best paper award).

7. G.C.H.E. de Croon, S. Nolfi, and E.O. Postma (2006). Towards pro-active embodied agents: on the importance of neural mechanisms suitable to process time information. *Complex Engineered Systems: Science Meets Technology (Understanding Complex Systems)* (eds. D. Braha, A. Minai, Y. Bar-Yam), pp. 338 - 363, Springer-Verlag, New York, NY.

8. G.C.H.E. de Croon and E.O. Postma (2006). Active object detection. *Belgian-Dutch AI Conference (BNAIC 2006), Namur, Belgium* (eds. P.-Y. Schobbens, W. Vanhoof, and G. Schwanen), pp. 107 – 114.

9. G.C.H.E. de Croon, E.O. Postma, and H.J. van den Herik (2006). A situated model for sensory-motor coordination in gaze control. *Pattern Recognition Letters*, Vol. 27, No. 11, pp. 1181 – 1190.

10. G.C.H.E. de Croon, I.G. Sprinkhuizen-Kuyper, and E.O. Postma (2006). Comparing active vision models. *MICC-IKAT Technical Report 06-02*, Universiteit Maastricht, Maastricht, the Netherlands.

11. G.C.H.E. de Croon and E.O. Postma (2007). Sensory-motor coordination in object detection. *1st IEEE Symposium on Artificial Life, Honolulu, HI* (ed. H.A. Abbass), pp. 147 – 154.

12. G.C.H.E. de Croon (2007). Active object detection. *2nd International Conference on Computer Vision Theory and Applications (VISAPP 2007), Barcelona, Spain* (eds. A. Ranchordas, H.J. Araújo, and J. Vitrià), pp. 97 – 103.

13. T. van Lankveld, G.C.H.E. de Croon, and K. Tuyls (2007). Static versus plastic controllers in evolutionary robotics. *Belgian-Dutch AI Conference (BNAIC 2007), Utrecht, the Netherlands* (eds. M. Dastani and E. de Jong), pp. 196 – 204.

14. G.C.H.E. de Croon, I.G. Sprinkhuizen-Kuyper, and E.O. Postma (in submission). Comparison of Active Vision Models. *Submitted to Image and Vision Computing*.

SIKS Dissertation Series

1998

1 Johan van den Akker (CWI[1]) *DEGAS - An Active, Temporal Database of Autonomous Objects*

2 Floris Wiesman (UM) *Information Retrieval by Graphically Browsing Meta-Information*

3 Ans Steuten (TUD) *A Contribution to the Linguistic Analysis of Business Conversations within the Language/Action Perspective*

4 Dennis Breuker (UM) *Memory versus Search in Games*

5 Eduard W. Oskamp (RUL) *Computerondersteuning bij Straftoemeting*

1999

1 Mark Sloof (VU) *Physiology of Quality Change Modelling; Automated Modelling of Quality Change of Agricultural Products*

2 Rob Potharst (EUR) *Classification using Decision Trees and Neural Nets*

3 Don Beal (UM) *The Nature of Minimax Search*

4 Jacques Penders (UM) *The Practical Art of Moving Physical Objects*

5 Aldo de Moor (KUB) *Empowering Communities: A Method for the Legitimate User-Driven Specification of Network Information Systems*

6 Niek J.E. Wijngaards (VU) *Re-Design of Compositional Systems*

7 David Spelt (UT) *Verification Support for Object Database Design*

8 Jacques H.J. Lenting (UM) *Informed Gambling: Conception and Analysis of a Multi-Agent Mechanism for Discrete Reallocation*

2000

1 Frank Niessink (VU) *Perspectives on Improving Software Maintenance*

2 Koen Holtman (TU/e) *Prototyping of CMS Storage Management*

3 Carolien M.T. Metselaar (UvA) *Sociaal-organisatorische Gevolgen van Kennistechnologie; een Procesbenadering en Actorperspectief*

[1]Abbreviations: SIKS – Dutch Research School for Information and Knowledge Systems; CWI – Centrum voor Wiskunde en Informatica, Amsterdam; EUR – Erasmus Universiteit, Rotterdam; KUB – Katholieke Universiteit Brabant, Tilburg; KUN – Katholieke Universiteit Nijmegen; RUL – Rijksuniversiteit Leiden; TUD – Technische Universiteit Delft; TU/e – Technische Universiteit Eindhoven; UL – Universiteit Leiden; UM – Universiteit Maastricht; UT – Universiteit Twente, Enschede; UU – Universiteit Utrecht; UvA – Universiteit van Amsterdam; UvT – Universiteit van Tilburg; VU – Vrije Universiteit, Amsterdam; RUN – Radboud Universiteit Nijmegen.

4 Geert de Haan (VU) *ETAG, A Formal Model of Competence Knowledge for User Interface Design*
5 Ruud van der Pol (UM) *Knowledge-based Query Formulation in Information Retrieval*
6 Rogier van Eijk (UU) *Programming Languages for Agent Communication*
7 Niels Peek (UU) *Decision-Theoretic Planning of Clinical Patient Management*
8 Veerle Coupé (EUR) *Sensitivity Analyis of Decision-Theoretic Networks*
9 Florian Waas (CWI) *Principles of Probabilistic Query Optimization*
10 Niels Nes (CWI) *Image Database Management System Design Considerations, Algorithms and Architecture*
11 Jonas Karlsson (CWI) *Scalable Distributed Data Structures for Database Management*

2001

1 Silja Renooij (UU) *Qualitative Approaches to Quantifying Probabilistic Networks*
2 Koen Hindriks (UU) *Agent Programming Languages: Programming with Mental Models*
3 Maarten van Someren (UvA) *Learning as Problem Solving*
4 Evgueni Smirnov (UM) *Conjunctive and Disjunctive Version Spaces with Instance-based Boundary Sets*
5 Jacco van Ossenbruggen (VU) *Processing Structured Hypermedia: A Matter of Style*
6 Martijn van Welie (VU) *Task-based User Interface Design*
7 Bastiaan Schonhage (VU) *Diva: Architectural Perspectives on Information Visualization*
8 Pascal van Eck (VU) *A Compositional Semantic Structure for Multi-Agent Systems Dynamics*
9 Pieter Jan 't Hoen (RUL) *Towards Distributed Development of Large Object-Oriented Models, Views of Packages as Classes*
10 Maarten Sierhuis (UvA) *Modeling and Simulating Work Practice BRAHMS: a Multiagent Modeling and Simulation Language for Work Practice Analysis and Design*
11 Tom M. van Engers (VU) *Knowledge Management: The Role of Mental Models in Business Systems Design*

2002

1 Nico Lassing (VU) *Architecture-Level Modifiability Analysis*
2 Roelof van Zwol (UT) *Modelling and Searching Web-based Document Collections*
3 Henk Ernst Blok (UT) *Database Optimization Aspects for Information Retrieval*
4 Juan Roberto Castelo Valdueza (UU) *The Discrete Acyclic Digraph Markov Model in Data Mining*
5 Radu Serban (VU) *The Private Cyberspace Modeling Electronic Environments Inhabited by Privacy-Concerned Agents*
6 Laurens Mommers (UL) *Applied Legal Epistemology; Building a Knowledge-based Ontology of the Legal Domain*
7 Peter Boncz (CWI) *Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications*
8 Jaap Gordijn (VU) *Value Based Requirements Engineering: Exploring Innovative E-Commerce Ideas*

9 Willem-Jan van den Heuvel (KUB) *Integrating Modern Business Applications with Objectified Legacy Systems*
10 Brian Sheppard (UM) *Towards Perfect Play of Scrabble*
11 Wouter C.A. Wijngaards (VU) *Agent Based Modelling of Dynamics: Biological and Organisational Applications*
12 Albrecht Schmidt (UvA) *Processing XML in Database Systems*
13 Hongjing Wu (TU/e) *A Reference Architecture for Adaptive Hypermedia Applications*
14 Wieke de Vries (UU) *Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems*
15 Rik Eshuis (UT) *Semantics and Verification of UML Activity Diagrams for Workflow Modelling*
16 Pieter van Langen (VU) *The Anatomy of Design: Foundations, Models and Applications*
17 Stefan Manegold (UvA) *Understanding, Modeling, and Improving Main-Memory Database Performance*

2003

1 Heiner Stuckenschmidt (VU) *Ontology-based Information Sharing in Weakly Structured Environments*
2 Jan Broersen (VU) *Modal Action Logics for Reasoning About Reactive Systems*
3 Martijn Schuemie (TUD) *Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy*
4 Milan Petković (UT) *Content-based Video Retrieval Supported by Database Technology*
5 Jos Lehmann (UvA) *Causation in Artificial Intelligence and Law – A Modelling Approach*
6 Boris van Schooten (UT) *Development and Specification of Virtual Environments*
7 Machiel Jansen (UvA) *Formal Explorations of Knowledge Intensive Tasks*
8 Yong-Ping Ran (UM) *Repair-based Scheduling*
9 Rens Kortmann (UM) *The Resolution of Visually Guided Behaviour*
10 Andreas Lincke (UT) *Electronic Business Negotiation: Some Experimental Studies on the Interaction between Medium, Innovation Context and Cult*
11 Simon Keizer (UT) *Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks*
12 Roeland Ordelman (UT) *Dutch Speech Recognition in Multimedia Information Retrieval*
13 Jeroen Donkers (UM) *Nosce Hostem – Searching with Opponent Models*
14 Stijn Hoppenbrouwers (KUN) *Freezing Language: Conceptualisation Processes across ICT-Supported Organisations*
15 Mathijs de Weerdt (TUD) *Plan Merging in Multi-Agent Systems*
16 Menzo Windhouwer (CWI) *Feature Grammar Systems - Incremental Maintenance of Indexes to Digital Media Warehouse*
17 David Jansen (UT) *Extensions of Statecharts with Probability, Time, and Stochastic Timing*
18 Levente Kocsis (UM) *Learning Search Decisions*

2004

1 Virginia Dignum (UU) *A Model for Organizational Interaction: Based on Agents, Founded in Logic*
2 Lai Xu (UvT) *Monitoring Multi-party Contracts for E-business*
3 Perry Groot (VU) *A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving*
4 Chris van Aart (UvA) *Organizational Principles for Multi-Agent Architectures*
5 Viara Popova (EUR) *Knowledge Discovery and Monotonicity*
6 Bart-Jan Hommes (TUD) *The Evaluation of Business Process Modeling Techniques*
7 Elise Boltjes (UM) *Voorbeeld$_{IG}$ Onderwijs; Voorbeeldgestuurd Onderwijs, een Opstap naar Abstract Denken, vooral voor Meisjes*
8 Joop Verbeek (UM) *Politie en de Nieuwe Internationale Informatiemarkt, Grensregionale Politiële Gegevensuitwisseling en Digitale Expertise*
9 Martin Caminada (VU) *For the Sake of the Argument; Explorations into Argument-based Reasoning*
10 Suzanne Kabel (UvA) *Knowledge-rich Indexing of Learning-objects*
11 Michel Klein (VU) *Change Management for Distributed Ontologies*
12 The Duy Bui (UT) *Creating Emotions and Facial Expressions for Embodied Agents*
13 Wojciech Jamroga (UT) *Using Multiple Models of Reality: On Agents who Know how to Play*
14 Paul Harrenstein (UU) *Logic in Conflict. Logical Explorations in Strategic Equilibrium*
15 Arno Knobbe (UU) *Multi-Relational Data Mining*
16 Federico Divina (VU) *Hybrid Genetic Relational Search for Inductive Learning*
17 Mark Winands (UM) *Informed Search in Complex Games*
18 Vania Bessa Machado (UvA) *Supporting the Construction of Qualitative Knowledge Models*
19 Thijs Westerveld (UT) *Using Generative Probabilistic Models for Multimedia Retrieval*
20 Madelon Evers (Nyenrode) *Learning from Design: Facilitating Multidisciplinary Design Teams*

2005

1 Floor Verdenius (UvA) *Methodological Aspects of Designing Induction-based Applications*
2 Erik van der Werf (UM) *AI Techniques for the Game of Go*
3 Franc Grootjen (RUN) *A Pragmatic Approach to the Conceptualisation of Language*
4 Nirvana Meratnia (UT) *Towards Database Support for Moving Object data*
5 Gabriel Infante-Lopez (UvA) *Two-Level Probabilistic Grammars for Natural Language Parsing*
6 Pieter Spronck (UM) *Adaptive Game AI*
7 Flavius Frasincar (TU/e) *Hypermedia Presentation Generation for Semantic Web Information Systems*
8 Richard Vdovjak (TU/e) *A Model-driven Approach for Building Distributed Ontology-based Web Applications*
9 Jeen Broekstra (VU) *Storage, Querying and Inferencing for Semantic Web Languages*

10 Anders Bouwer (UvA) *Explaining Behaviour: Using Qualitative Simulation in Interactive Learning Environments*
11 Elth Ogston (VU) *Agent Based Matchmaking and Clustering - A Decentralized Approach to Search*
12 Csaba Boer (EUR) *Distributed Simulation in Industry*
13 Fred Hamburg (UL) *Een Computermodel voor het Ondersteunen van Euthanasiebeslissingen*
14 Borys Omelayenko (VU) *Web-Service Configuration on the Semantic Web; Exploring how Semantics Meets Pragmatics*
15 Tibor Bosse (VU) *Analysis of the Dynamics of Cognitive Processes*
16 Joris Graaumans (UU) *Usability of XML Query Languages*
17 Boris Shishkov (TUD) *Software Specification Based on Re-usable Business Components*
18 Danielle Sent (UU) *Test-selection Strategies for Probabilistic Networks*
19 Michel van Dartel (UM) *Situated Representation*
20 Cristina Coteanu (UL) *Cyber Consumer Law, State of the Art and Perspectives*
21 Wijnand Derks (UT) *Improving Concurrency and Recovery in Database Systems by Exploiting Application Semantics*

2006

1 Samuil Angelov (TU/e) *Foundations of B2B Electronic Contracting*
2 Cristina Chisalita (VU) *Contextual Issues in the Design and Use of Information Technology in Organizations*
3 Noor Christoph (UvA) *The Role of Metacognitive Skills in Learning to Solve Problems*
4 Marta Sabou (VU) *Building Web Service Ontologies*
5 Cees Pierik (UU) *Validation Techniques for Object-Oriented Proof Outlines*
6 Ziv Baida (VU) *Software-aided Service Bundling - Intelligent Methods & Tools for Graphical Service Modeling*
7 Marko Smiljanic (UT) *XML Schema Matching – Balancing Efficiency and Effectiveness by means of Clustering*
8 Eelco Herder (UT) *Forward, Back and Home Again - Analyzing User Behavior on the Web*
9 Mohamed Wahdan (UM) *Automatic Formulation of the Auditor's Opinion*
10 Ronny Siebes (VU) *Semantic Routing in Peer-to-Peer Systems*
11 Joeri van Ruth (UT) *Flattening Queries over Nested Data Types*
12 Bert Bongers (VU) *Interactivation - Towards an E-cology of People, our Technological Environment, and the Arts*
13 Henk-Jan Lebbink (UU) *Dialogue and Decision Games for Information Exchanging Agents*
14 Johan Hoorn (VU) *Software Requirements: Update, Upgrade, Redesign - towards a Theory of Requirements Change*
15 Rainer Malik (UU) *CONAN: Text Mining in the Biomedical Domain*
16 Carsten Riggelsen (UU) *Approximation Methods for Efficient Learning of Bayesian Networks*
17 Stacey Nagata (UU) *User Assistance for Multitasking with Interruptions on a Mobile Device*

18 Valentin Zhizhkun (UvA) *Graph Transformation for Natural Language Processing*
19 Birna van Riemsdijk (UU) *Cognitive Agent Programming: A Semantic Approach*
20 Marina Velikova (UvT) *Monotone Models for Prediction in Data Mining*
21 Bas van Gils (RUN) *Aptness on the Web*
22 Paul de Vrieze (RUN) *Fundaments of Adaptive Personalisation*
23 Ion Juvina (UU) *Development of Cognitive Model for Navigating on the Web*
24 Laura Hollink (VU) *Semantic Annotation for Retrieval of Visual Resources*
25 Madalina Drugan (UU) *Conditional log-likelihood MDL and Evolutionary MCMC*
26 Vojkan Mihajlović (UT) *Score Region Algebra: A Flexible Framework for Structured Information Retrieval*
27 Stefano Bocconi (CWI) *Vox Populi: Generating Video Documentaries from Semantically Annotated Media Repositories*
28 Borkur Sigurbjornsson (UvA) *Focused Information Access using XML Element Retrieval*

2007

1 Kees Leune (UvT) *Access Control and Service-Oriented Architectures*
2 Wouter Teepe (RUG) *Reconciling Information Exchange and Confidentiality: A Formal Approach*
3 Peter Mika (VU) *Social Networks and the Semantic Web*
4 Jurriaan van Diggelen (UU) *Achieving Semantic Interoperability in Multi-agent Systems: a Dialogue-based Approach*
5 Bart Schermer (UL) *Software Agents, Surveillance, and the Right to Privacy: a Legislative Framework for Agent-enabled Surveillance*
6 Gilad Mishne (UvA) *Applied Text Analytics for Blogs*
7 Natasa Jovanovic' (UT) *To Whom It May Concern - Addressee Identification in Face-to-Face Meetings*
8 Mark Hoogendoorn (VU) *Modeling of Change in Multi-Agent Organizations*
9 David Mobach (VU) *Agent-based Mediated Service Negotiation*
10 Huib Aldewereld (UU) *Autonomy vs. Conformity: an Institutional Perspective on Norms and Protocols*
11 Natalia Stash (TU/e) *Incorporating Cognitive/Learning Styles in a General-Purpose Adaptive Hypermedia System*
12 Marcel van Gerven (RUN) *Bayesian Networks for Clinical Decision Support: A Rational Approach to Dynamic Decision-Making under Uncertainty*
13 Rutger Rienks (UT) *Meetings in Smart Environments; Implications of Progressing Technology*
14 Niek Bergboer (UM) *Context-based Image Analysis*
15 Joyca Lacroix (UM) *NIM: a Situated Computational Memory Model*
16 Davide Grossi (UU) *Designing Invisible Handcuffs. Formal investigations in Institutions and Organizations for Multi-agent Systems*
17 Theodore Charitos (UU) *Reasoning with Dynamic Networks in Practice*
18 Bart Orriens (UvT) *On the Development and Management of Adaptive Business Collaborations*
19 David Levy (UM) *Intimate Relationships with Artificial Partners*

20 Slinger Jansen (UU) *Customer Configuration Updating in a Software Supply Network*
21 Karianne Vermaas (UU) *Fast Diffusion and Broadening Use: a Research on Residential Adoption and Usage of Broadband Internet in the Netherlands between 2001 and 2005*
22 Zlatko Zlatev (UT) *Goal-oriented Design of Value and Process Models from Patterns*
23 Peter Barna (TU/e) *Specification of Application Logic in Web Information Systems*
24 Georgina Ramrez Camps (CWI) *Structural Features in XML Retrieval*
25 Joost Schalken (VU) *Empirical Investigations in Software Process Improvement*

2008

1 Katalin Boer-Sorbán (EUR) *Agent-based Simulation of Financial Markets: a Modular, Continuous-time Approach*
2 Alexei Sharpanskykh (VU) *On Computer-Aided Methods for Modeling and Analysis of Organizations*
3 Vera Hollink (UvA) *Optimizing Hierarchical Menus: a Usage-based Approach*
4 Ander de Keijzer (UT) *Management of Uncertain Data - towards Unattended Integration*
5 Bela Mutschler (UT) *Modeling and Simulating Causal Dependencies on Process-aware Information Systems from a Cost Perspective*
6 Arjen Hommersom (RUN) *On the Application of Formal Methods to Clinical Guidelines, an Artificial Intelligence Perspective*
7 Peter van Rosmalen (OU) *Supporting the Tutor in the Design and Support of Adaptive E-learning*
8 Janneke Bolt (UU) *Bayesian Networks: Aspects of Approximate Inference*
9 Christof van Nimwegen (UU) *The Paradox of the Guided User: Assistance can be Counter-effective*
10 Wauter Bosma (UT) *Discourse Oriented Summarization*
11 Vera Kartseva (VU) *Designing Controls for Network Organizations: a Value-Based Approach*
12 Jozsef Farkas (RUN) *A Semiotically Oriented Cognitive Model of Knowledge Representation*
13 Caterina Carraciolo (UvA) *Topic Driven Access to Scientific Handbooks*
14 Arthur van Bunningen (UT) *Context-Aware Querying; Better Answers with Less Effort*
15 Martijn van Otterlo (UT) *The Logic of Adaptive Behavior: Knowledge Representation and Algorithms for the Markov Decision Process Framework in First-Order Domains*
16 Henriette van Vugt (VU) *Embodied Agents from a User's Perspective*
17 Martin Op 't Land (TUD) *Applying Architecture and Ontology to the Splitting and Allying of Enterprises*